1991

Dynamic scheduling strategies for pseudo parallel queues: Observing queues before joining.

Wai-Hang. Poon
University of Windsor

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

DYNAMIC SCHEDULING STRATEGIES FOR PSEUDO PARALLEL QUEUES — "OBSERVING QUEUES BEFORE JOINING"

by

Wai-Hang Poon

A Thesis
Submitted to the Faculty of Graduate Studies and Research
through the School of Computer Science in Partial
Fulfillment of the Requirements for the Degree of
Master of Computer Science at the
University of Windsor

Windsor, Ontario, Canada
1991

ISBN 0-315-72804-3

Abstract

In many real world systems, servers are assigned to work in parallel to increase the system throughput and resource utilization. A class of multiple server queues, called pseudo parallel queues, was investigated. Exponential service times and interarrival times are assumed throughout.

We are especially interested in establishing strategies for special customers to lower their sojourn time. Regular customers are assumed to join the shortest queue (JSQ) among the parallel queues. Decisions of the regular customers are made at their arrival time.

The only information available to a special customer is the current queue lengths of the parallel queues at and after the time it arrives at the system. Two job scheduling strategies for the special customers are proposed based on a "gathering information" philosophy. They are studied for pseudo parallel queues.

The thesis statement is that *under certain circumstances, the average sojourn time of an individual job can be shortened by gathering information about the pseudo parallel queues before joining the queue.*

The purpose of this thesis is:

1. to provide a comprehensive literature review of multiple server queues,
2. to explore some mathematical properties of the parallel queues (by assuming JSQ),
3. to use the properties of the parallel queues that we found in analyzing new scheduling strategies.

To my parents.

Acknowledgments

I would like to thank Dr. Hlynka for his help, his direction, and his great effort in this thesis. Valuable suggestions have been provided by my committee members, Dr. Toews, Dr. Bandyopadhyay, Dr. Caron. Suggestions were also given by Dr. Frost to improve the survey.

This work is supported by grants from NSERC (Natural Sciences and Engineering Research Council of Canada) and an assistantship from the School of Computer Science at the University of Windsor. Special thanks to my family for supporting me.

I would also like to thank all those who helped me to enjoyed the university life and experience.

My life in Canada has been changed since I entered this University. Thanks to Dr. Atkinson, the former associate dean of science, for admitting me.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER 1 INTRODUCTION

We often find a number of parallel queues of customers waiting for service. We want to join a queue that will let us leave the system as soon as possible. One example is a supermarket where customers line up in parallel to pay for their goods. The join-queue decision is of interest in this thesis. In this chapter, we give an overview of the problem studied. Details of the problem will be introduced in next chapter.

In this chapter, basic elements of the problem such as queues, queueing theory, and pseudo parallel queues will be introduced. The thesis statement is also presented. The layout of this thesis will be shown at the end of this chapter.

1.1 Queues

Different kinds of queueing systems can be found throughout the real world. Some examples are automobile traffic jams, customers lining up in front of check out stations, people lining up in a bank, jobs stacking up on a secretary's desk, and print jobs queueing in the print buffer.

A queueing system can be considered as a set of items that allows objects to enter, stay, and leave the set according to the objects' behavior, see figure 10.

The processes of objects entering and leaving the system are called the arrival process and the departure process, respectively. The objects flowing through the queueing system could be jobs, customers, cars or positions on a conveyor belt. The terms jobs and customers will be used interchangeably for the objects.

The arrival and departure processes can depend on anything, including time and the load of the system. The interarrival times and service times can follow various distributions, with various types of dependence.

1.2 Queueing Theory

Queueing theory is a tool that can be used to predict the performance and assist in designing procedures of queueing systems. It has been developing since the mid-1930's. Its earliest application was in telephone engineering. It is now widely used by many industries including the computer industry.

Typical steps of analyzing real world problems in terms of queueing theory are the following:

1. Model the real world problem with a queueing model. The real world system could be a transportation system, a food store, or a bank.
2. The queueing model is then simplified by some assumptions.
3. The queueing model is analyzed by mathematical techniques and simulation. Properties of the model are discovered.
4. If he properties are unreasonable, then we refine the assumptions (go to step 3).
5. Solve the original problem using the properties given in the analysis step.

1.3 Pseudo Parallel Queues[1]

In many systems, multiple servers are assigned to work in parallel to increase the system throughput and the utilization of all existing resources. Multiple server queueing systems, as shown in figure (1), are classified into parallel queues and combined queueing systems as discussed in appendix A.

---

[1] NOTE: Details of the terms used in this section will be introduced in Appendix A.

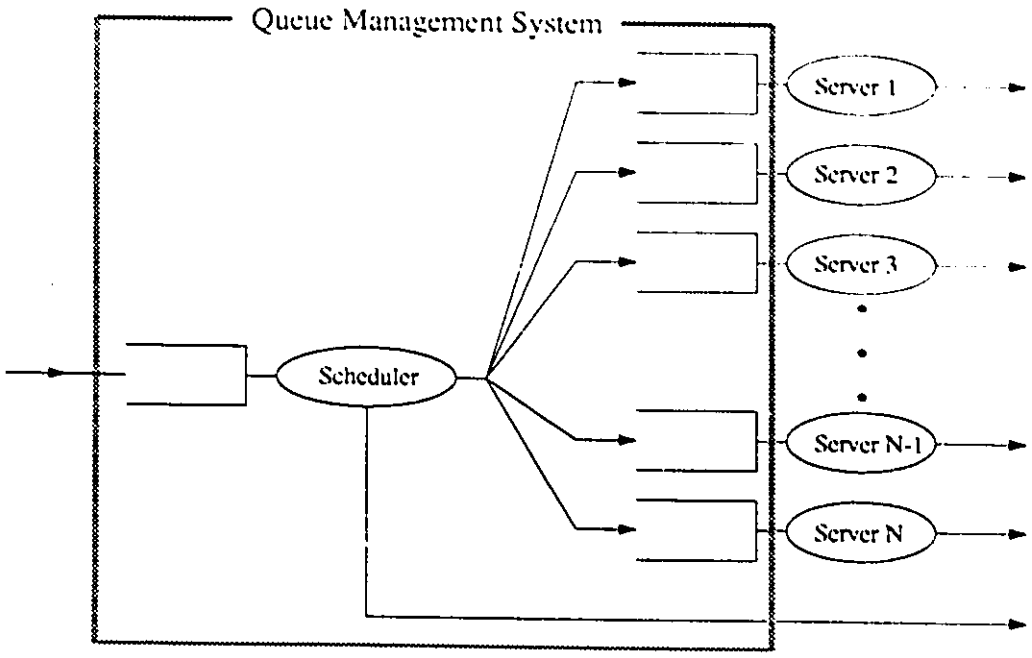

Figure 1 The General Multiple Server Queueing System

Pseudo parallel queues are the focus of this thesis. This system is essentially a general multiple server queueing system except that the size of the shared buffer is only one and each server is atomic[2]. The customers have to pass through the shared buffer before they join any parallel queue in the system. Arriving customers are free to stay in or pass through the common buffer immediately. Any customer can bypass other customers that decide to stay in the buffer. However, jockeying, or moving from one queue to another, is prohibited.

We divide customers into two types — regular customers and a single special (smart) customer. Regular customers are assumed to pass through the buffer immediately and join the shortest of the parallel queues. Various join-queue strategies for the smart customer were studied. The objective of the smart customer is to minimize its sojourn time. The information available to the smart customer at the time it arrives to the system is the lengths of the parallel queues.

[2] An atomic server means a single server. In other words, the "server" is not a team of servers.

Three job scheduling strategies for the smart customer were proposed and studied using analytical techniques and simulation.

The thesis shows that *under certain circumstances, the average sojourn time of an individual job can be shortened by gathering information about the pseudo parallel queues before joining the queue.*

1.4 Organization of the Thesis

This thesis is divided into five major chapters -- Introduction, Motivation, Properties of Parallel Queues, Observing Queues Before Joining and Conclusion. Chapters 3 and 4 are the core of this thesis.

The second chapter will motivate the study of pseudo parallel queues with a few real world examples. Then the pseudo parallel queueing model will be formally defined. The assumptions of the model and their validity will be discussed briefly.

The third chapter will discuss some basic properties of parallel queues. This will include a brief review of research in the area and an investigation of new results.

The fourth chapter, observing queues before joining, is the main focus of this thesis. We will propose two join-queue strategies. Performance of these strategies will be analyzed by using the analytical results obtained in Chapter 3.

The last chapter will summarize the findings of the thesis, discuss the relationship between these findings and known results, and propose possible improvements to the queueing model as future work.

A comprehensive literature survey of multiple server queues is provided in appendix A.

CHAPTER 2 MOTIVATION OF THE THESIS

Two real world systems will be discussed to motivate the importance of the thesis. The importance of the results is not limited to these two examples, since the results also apply to other systems that can be modeled with the same model.

2.1 Real World System I — Grocery Supplier's Decision

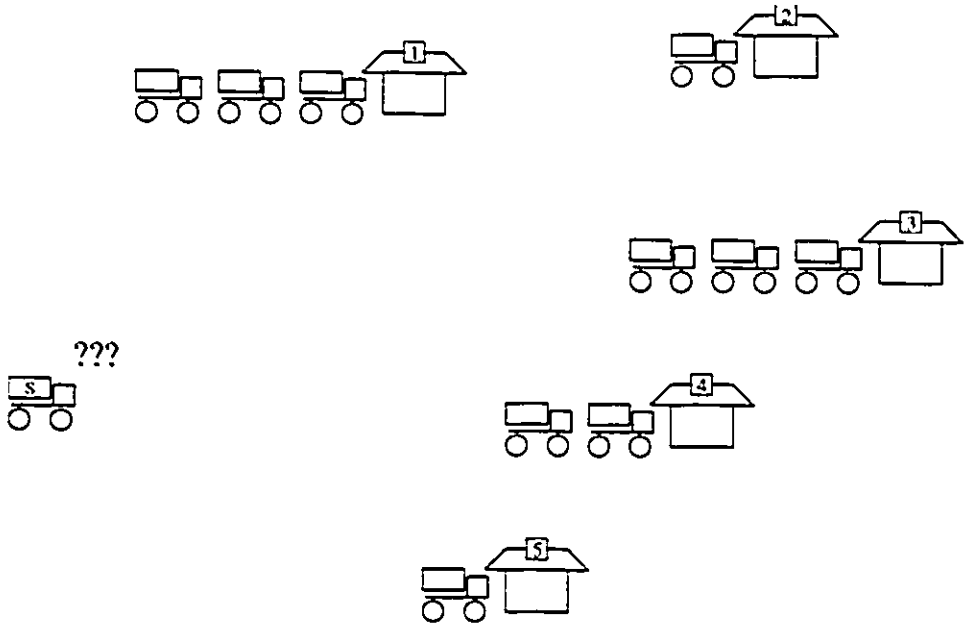

Figure 2 Grocery Supplier's Decision

Suppose a manufacturer has a batch of products such as groceries. There are several stores to which the manufacturer can sell the product, as shown in figure 2. The manufacturer receives no money until the batch of product is completely sold. Thus the problem is to sell off the product as soon as possible in order to minimize the time to receive a return on the investment. This problem is even more important to suppliers of perishable inventory (such as agricultural products).

The suppliers should consider the amount of inventory and the selling-rate of every store, if the information is available. However, the amount of inventory and the selling-rate are usually difficult to obtain, especially for a new supplier.

By assuming that each supplier does not supply more than one store simultaneously, this system can be modelled by a pseudo parallel queues. The other basic assumptions are as follows:

1. All stores sell their inventory independently.
2. Any supplier can send his/her product to any store. The products supplied will eventually be sold out.
3. For each store, information available to the suppliers is only the number of suppliers that supply products to it. The information is always up-to-date[3].
4. Each supplier does not supply additional inventory to a store until the inventory supplied to the store by him/her is sold out.

One possible strategy to achieve the goal is to deliver the product to the store that has the least number of suppliers.

For example, there are currently three suppliers for store #1, one for store #2, three for store #3, two for store #4, and one for store #5. The new supplier should probably place his/her product in either store #2 or #5, assuming that the selling-rate of all stores, is almost the same.

A study of this system could:

1. Reduce costs due to the decay of perishable inventory and interest costs.
2. Shorten the term of investment and therefore make a faster profit.

---

[3] In an extremely dynamic market, the information the suppliers get is always not up-to-date.

2.2 Real World System II — Distributed Computing Systems

The speed of computation has increased since parallel computers were introduced. The multiprocessor system is the most common architecture for parallel computers today. However, it can be shown mathematically that multiprocessor systems are not scalable [Hwang84]. In the other words, the speedup of a multiprocessor system is not proportional to the number of processors. The reason is resource contention (for example, data structure contention). Because the processors share a set of data structure, they have to synchronize at all times. Therefore multiprocessor systems are not scalable.

A multicomputer system (distributed parallel computer system) is another architecture for parallel computers. It is also called a loosely coupled system, a distributed memory system, or a distributed computing system. The system is composed of a set of autonomous computing units. Each computing unit has its local memory. The computing units do not share any global memory. They communicate asynchronously by passing messages around the system. Therefore, the resource contention problem is reduced. Multicomputer systems are scalable.

A multicomputer system should be more efficient than a multiprocessor system, if the coordination among the computing units is good and the load is evenly distributed among the computing units. The difference between these two systems are summarized in table 1.

Multicomputer systems are not popular today, because of the difficulty of developing efficient software for them. "Software remains the final hurdle to clear if parallel processing via multicomputers is to emerge as a popular alternative to sequential processing" [Stein91]. New multicomputer programming paradigms for a message passing computer system are currently under research. Once the problem is solved, the future

of multicomputer systems is bright.

	Multiprocessor Systems	Multicomputer Systems
Communication	Takes place through the global variables in the shared memory	By passing messages around the system through high-speed communication lines such as buses and high-speed network.
Speedup	bounded by $O(\frac{n}{\log n})$	O(n)
Memory	Shared memory is accessible by all processors	No shared memory
Software Development	Similar to that for a traditional uniprocessor machine	Programmers have to take care of logical-concurrency, data-decomposition, and load-balancing [Stein91]

Table 1 Multiprocessor Systems vs Multicomputer Systems

There are two primary objectives of employing parallel architectures. The first is increasing system throughput. The second is speeding up a particular job. The first objective is, in our view, more important than the second for multiuser systems.

Suppose an application can be decomposed into two parallel tasks and there are two computing units in the multicomputer system. Since each computing unit has its own local memory and we intend to minimize the shared data structure, each computing unit has its own job queue. If the system found that there are ten tasks on the first computing unit and if there is one task executing on the second computing unit, how should these two tasks be assigned to the computing units? In order to balance the load among the computing units in the system, both new tasks should be assigned to the second computing unit, because the load of the second computing unit is not as heavy as the first one.

We consider a distributed system, as shown in figure 3, which is composed of a number of heterogeneous computing units such as work stations, mini-computers, and transputers. Each computing unit generates jobs from time to time and sends the job request to the poll of computing units. The jobs will be assigned to any computing unit (according to some strategy) and will be processed there.

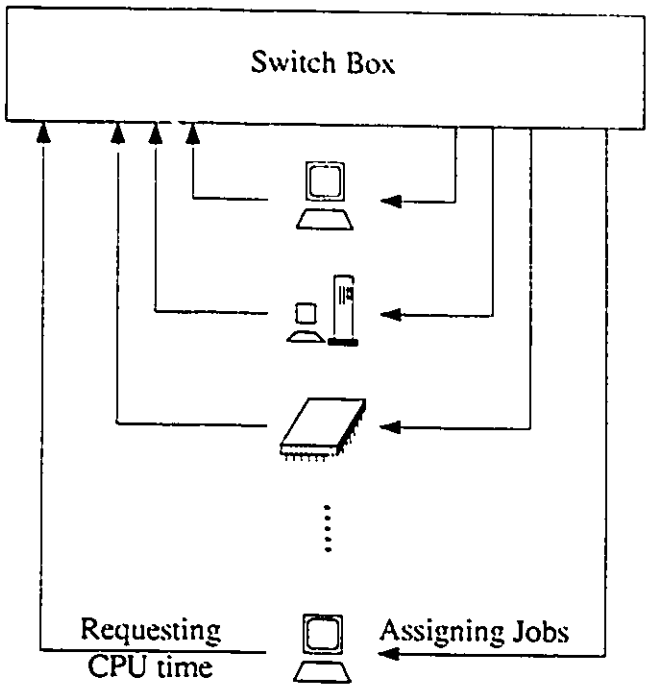

Figure 3 Job Scheduling of a Distributed System

To model the system, we consider each computing unit in the multicomputer system to be a server working independently of the other servers[4]. The computing jobs can be viewed as the customers of a queueing system. Studying this system could help in the design of distributed operating systems.

We assume the following:

1. All jobs are executed independently.

[4] This assumption ignores the fact that the computing units may communicate to each other to accomplish the same goal. This happens, if a thread has a number of tasks or the operating system decides to synchronize.

2. Any job can be processed by any computing unit. Computing units differ only in processing speed.
3. Computing units can communicate with each other extremely fast, so that the current lengths of job queues are known to all computing units and processes are ready to be executed immediately after being scheduling.

Once an regular job is created, the job must be dispatched to one of the computing units. As mentioned before, the load should be evenly distributed to all computing units. This load sharing problem has been referred to as the global scheduling subproblem [Wang85]. A typical way to distribute the load evenly to all the computing units is using the JSQ policy [Weber78].

The problem is to design a scheduling strategy for higher priority jobs that will shorten the system time of the higher priority jobs and share the resources fairly with regular jobs.

One method is preemption. However, this is not fair to regular jobs. In addition, the priority of the higher priority jobs is not as high as that of the memory manager and the file manager.

2.3 The Queueing Model under Investigation

The systems discussed in the last two sections share some properties:

1. There are multiple servers (stores and computing units).
2. There are parallel queues (room in the stores and job queues).
3. Each server is responsible for its own queue. (Each store sells its own inventory and each computing unit processes only the jobs in its own job queue).
4. The arrival process of customers (suppliers and jobs) is random.
5. The time to serve customers (selling grocery and processing jobs) is random.

A queueing model is an abstraction of a real world system. The advantage of studying the model is that we will understand all similar systems, if we understand the model. Moreover, problem of the same sort can be solved at the same time.

Pseudo parallel queues are essentially general multiple server queues with a shared small buffer. They have multiple servers working in parallel and each server has its own queue. Regular customers must join one of the queues at the instant of arrival. The queue sizes are arbitrary. Any customer can be served by any server. The service rates of the servers may differ.

	Jobs	Customers
Regular	The scheduler dispatches them with an ordinary strategy.	They do not know the smart strategy. They only apply the ordinary strategy.
Smart	The scheduler dispatches them with the smart strategy.	They are smarter than ordinary customers. They apply the smart strategy.

Table 2 Duties of the Scheduler

Much research has been done on control of queues but only a few articles focus on the individual objective control. We are interested in minimizing the sojourn time of an particular individual. The individual will be called the smart customer.

In the two examples considered earlier, the smart customer can be considered as

1. a smart new supplier in the supplier's decision example,
2. a higher priority job in the distributed computing system example.

Table 2 explains the relationships among the terms "jobs", "customers", "regular", "smart" and "job scheduler"[5].

[5] It is also referred to as job dispatcher and packet router in some articles.

If the same strategy is applied, the effect is the same whether the decision is made by the scheduler or the customer.

Open problem: What strategy will minimize the sojourn time of the smart customer?

2.4 Assumptions

The open problem is simple to state, but the solution is complicated. The appropriate strategy depends on the information available. The following assumptions are made about the queueing structure, the queueing discipline, and the information available.

1. Two parallel queues — Some properties obtained for two parallel queues can be extended to multiple parallel queues.
2. Infinite queue sizes — Unless specified otherwise.
3. Exponential service time and interarrival time with rates μ and λ respectively — If no information about the statistical distributions is available, these assumptions are valid by entropy theory. They are also the ordinary assumptions in queueing theory.
4. First come first serve queueing discipline (FCFS) — This is the discipline of queues in the grocery supplier example. A computer system usually applies a round-robin queueing discipline (RR)[6]. It can be proved that the steady state probabilities for the states of a two parallel queues that employs FCFS and PS are the same (with the method provided in [Sauer81]), provided that the interarrival distribution is exponential and the context switching overhead is very small.
5. Regular customers apply the "join the shortest queue" strategy (JSQ). This is usual human behavior as observed, for instance, in a supermarket. It has also been shown that the JSQ scheduling strategy is good for load balancing [Wang85, Boel89].

---

[6] Its ideal approximation is processor sharing (PS).

6. Jockeying is not allowed — Jockeying is difficult or perhaps impossible for some systems, such as queues of cars.
7. The average service rate and the average arrival rate are *unknown* to the customers.
8. The current lengths of all parallel queues are *known*.

2.5 Outline of Solution Procedure and Expected Result

A partial solution to the problem is a scheduling strategy that makes the average sojourn time of the smart customer shorter than that of the regular customers. It is difficult to define the feasible solution space for the problem of finding the optimal strategy. It is even more difficult to find the optimal solution.

We propose two strategies based on the heuristic that "gathering information is the first rule of winners." We compare the average sojourn time of an arriving customer under two proposed strategies and the JSQ strategy. These strategies are all dynamic strategies[7]. We apply both analytical techniques and simulation for the comparison.

The solution procedure is as follows:

1. Study the properties of two parallel queues, by assuming the JSQ scheduling strategy and no smart customer. (Chapter 3)
2. Compare the performance of strategies 1, 2, and 3 analytically. (Chapter 4)
3. Compare the performance of strategies 1, 2, and 3 by using simulation. The purpose of this step is to verify the analytical result. (Chapter 4)

Since strategies 2 and 3 require information to be collected, we expect strategies 2 and 3 will do better than strategy 1 in some situations. If this is true, the heuristic "gathering information is the first rule of winners" can help to define the feasible solution space of the problem.

---

[7] Since dynamic strategies are better than static [Ephremides80], we start our optimal solution search with dynamic strategies.

CHAPTER 3 PROPERTIES OF THE STEADY STATE PROBABILITIES OF TWO PARALLEL QUEUES

In this chapter, we will analyze a queueing system with two servers, each with its own queue of infinite size. The interarrival times are assumed to be exponentially distributed. The service time for each server is exponentially distributed but the rates may be different. No jockeying between the two queues is allowed[8].

Arriving customers always join the shortest queue. If the queues have the same size, a customer picks one of the two queues at random to join with probability 0.5 . In fast food industries (eg. McDonald's), transportation (eg. toll booths), computer technology (eg. the global scheduling subproblem in distributed systems and the hot potato algorithm for computer networks), and many other situations, customers are often confronted with several queues, each with its own server. Customers generally enter the shortest queue available.

We attempted to obtain some properties of steady state probabilities of two parallel queues with four elementary approaches. Two of the approaches are quite successful and a number of properties are obtained. In Chapter 4, the three strategies defined in the previous chapter will be analyzed with these properties.

3.1 Early Research of Two Parallel Queues

The study of parallel queues was initiated by Haight in 1958. His work was followed by Kingman (1961), and others. We define the states of a two parallel queues in section 3.2. Major efforts have been made to obtain the steady state probability distribution of the parallel queues, where the steady state probability or limiting probability is the long

[8] In some situations, customers move from one queue to another. This movement is called "jockeying". In other situations, such as when a customer is a car, jockeying is difficult or perhaps impossible.

run probability of being in a particular state (i.e. the proportion of time that the sytem spends in that particular state). More than ten papers have been published with this goal. Some of the general approaches are listed below:

1. Find bounds on probabilities from the rate matrix,
2. Approximate probability distributions directly from the rate matrix or from generating functions,
3. Find the asymptotic probability distribution for the limiting case, and
4. Approximate the probability distribution by comparing the queueing system with other similar queueing systems.

Haight (1958) applied generating functions to obtain some properties of the system. He also investigated the system with jockeying allowed.

Kingman (1961) assumed that a customer would join the shortest queue (JSQ) available. If both queues were the same size then a customer would join either queue with probability 0.5 . This JSQ strategy is also referred to as the symmetric JSQ. Haight studied the asymmetric JSQ systems where in the case of equal queue lengths, the probability of joining the first and second queues are 1 and 0 respectively. Kingman assumed that both servers had the same service rate. The problem analyzed by Kingman is generally referred to as the "shortest queue problem."

Flatto and McKean (1977) used complex analysis to approximate the limiting probabilities in the case where the two service rates were equal. An approximation was made for the limiting case where lengths of the queues are large.

Halfin (1985) found bounds on the limiting probabilities of the system having exactly n customers. Once again it was assumed that the two service rates were equal. The results were derived from the state transition rate diagram.

Gubner, Gopinath and Varadhan (1989) showed that the value $E(|L_i(t) - L_j(t)| \mid L(0) = \underline{1})$ is bounded under the JSQ strategy, where $\underline{1}$ is the initial state vector and $L_i(t)$ is length of the i^{th} queue at time $t = 0, 1, 2, \ldots$ and $i \neq j$. This is true even under arbitrarily heavy traffic [Gubner89].

Some research attempted to approximate the steady state probability distribution from the knowledge of either the state transition rate diagram of the original system or of other similar systems.

Hall and Disney (1971) studied a similar system with a semi-Markov arrival process, finite size parallel queues, and heterogenous service rates.

Grassmann (1980) used truncation of the queues to obtain numerical results.

Conolly (1984) studied the system with finite size queues and compared it to other similar systems.

Gertsbakh (1984) analyzed the shortest queue problem for the case when the two servers have equal rates, using the matrix geometric methods of Neuts (1981).

Knessl, Matkowsky, Schuss and Tier (1986) gave asymptotic evaluations of the steady state probability distribution for the heterogenous service rate case.

Rao and Posner (1987) found the probability distribution of the total number of customers in the system for the heterogenous service rate case, when the size of one of the queues is finite.

Blanc (1987) proposed a method based on power series expansions and recursion to approximate the steady state probability and some moments. The method originated from approximating the global balance equations of the steady state probabilities. His study was for systems with multiple servers.

Nelson and Philips (1989) derived an approximation for the mean response time of the parallel queues with multiple servers.

Zhao and Grassman (1989) derived formulae for homogenous servers based on Flatto and McKean (1977). The new formulae which were superior for numerical purposes.

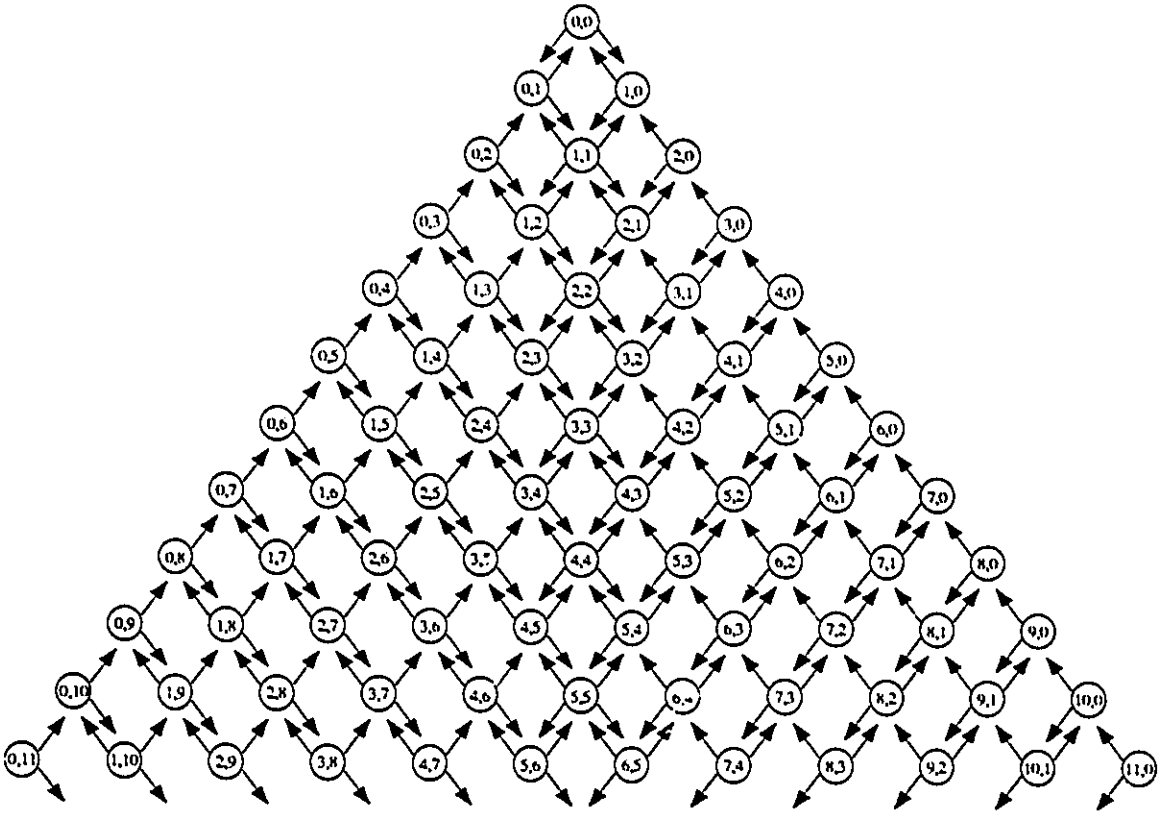

Figure 4 State Transition Rate Diagram of a Two Parallel Queues

In this chapter, we will introduce new relationships between the limiting probabilities using simple techniques. We will also introduce some useful bounds on certain probabilities, which turn out to be useful in comparing joining strategies for two parallel queues. Our assumptions are those of Kingman but we allow the servers to have different service rates. We assume exponentially distributed interarrival times and service times. We assume two servers. We assume no jockeying between queues. Like Kingman, we

assume symmetric JSQ. We call this queueing system A. In Section 3.4 and 3.5, we will introduce systems B(r) and C(r).

λ	$\lambda/2$	$\lambda/2$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
μ_2	B	0	0	λ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
μ_1	0	A	0	λ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...
0	μ_2	0	B	0	0	0	λ	0	0	0	0	0	0	0	0	0	0	0	0	0	...
0	μ_1	μ_2	0	C	0	0	$\lambda/2$	$\lambda/2$	0	0	0	0	0	0	0	0	0	0	0	0	...
0	0	μ_1	0	0	A	0	0	λ	0	0	0	0	0	0	0	0	0	0	0	0	...
0	0	0	μ_2	0	0	B	0	0	0	0	λ	0	0	0	0	0	0	0	0	0	...
0	0	0	μ_1	μ_2	0	0	C	0	0	0	0	λ	0	0	0	0	0	0	0	0	...
0	0	0	0	μ_1	μ_2	0	0	C	0	0	0	λ	0	0	0	0	0	0	0	0	...
0	0	0	0	0	μ_1	0	0	0	A	0	0	0	λ	0	0	0	0	0	0	0	...
0	0	0	0	0	0	μ_2	0	0	0	B	0	0	0	0	0	λ	0	0	0	0	...
0	0	0	0	0	0	μ_1	μ_2	0	0	0	C	0	0	0	0	0	λ	0	0	0	...
0	0	0	0	0	0	0	μ_1	μ_2	0	0	0	C	0	0	0	0	$\lambda/2$	$\lambda/2$	0	0	...
0	0	0	0	0	0	0	0	μ_1	μ_2	0	0	0	C	0	0	0	0	λ	0	0	...
0	0	0	0	0	0	0	0	0	μ_1	0	0	0	0	A	0	0	0	0	λ	0	...
0	0	0	0	0	0	0	0	0	0	μ_2	0	0	0	0	B	0	0	0	0	0	...
0	0	0	0	0	0	0	0	0	0	μ_1	μ_2	0	0	0	0	C	0	0	0	0	...
0	0	0	0	G	0	0	0	0	0	0	μ_1	μ_2	0	0	0	0	C	0	0	0	...
0	0	0	0	0	0	0	0	0	0	0	0	μ_1	μ_2	0	0	0	0	C	0	0	...
0	0	0	0	0	0	0	0	0	0	0	0	0	μ_1	μ_2	0	0	0	0	C	0	...
0	0	0	0	0	0	0	0	0	0	0	0	0	0	μ_1	0	0	0	0	0	A	...

Figure 5 Rate Matrix of the Two Parallel Queues

3.2 Notation

Assume customers arrive at rate λ and enter the shortest queue. The two servers have service rates μ_1 and μ_2. The states of the system are of the form (i, j), where the first component of the ordered pair represents the number of customers in the queue of the first server and the second component represents the number of customers in the queue of the second server. Figure 4 shows the state transition rate diagram of the two parallel server queues. Each arrow pointing to the top left hand corner of the paper has rate μ_1. Each arrow pointing to the top right hand corner has rate μ_2. All arrows pointing from the states $\{(i, j) \mid i = j\}$ to the bottom left and right corners have rates $\lambda/2$. All other arrows pointing downward have rates λ.

A rate matrix Λ can be constructed for a Markov process. Each element $\lambda_{s1, s2}$ of the rate matrix Λ is the rate of change from state s1 to state s2, if s1 does not equal to s2. $\lambda_{s1, s1}$ is chosen so that each row of the rate matrix sums to zero.

Let $A = -(\mu_1 + \lambda)$, $B = -(\mu_2 + \lambda)$, and $C = -(\mu_1 + \mu_2 + \lambda)$. The states of system A are listed in the order of (0,0), (0,1), (1,0), (0,2), (1,1), (2,0), . . . If we group all states as $\{(i, j) \mid i + j = n\}$ for $n = 0, 1, 2, \ldots$, then the rate matrix Λ has a block form[9] as shown in Figure 5. Each column of the matrix corresponds to an global balance equation of each of the states (0,0), (0,1), (1,0), (0,2), (1,1), (2,0) . . . , as discussed in the next section.

3.3 Analysis Approach I: Balance Flow Rates

Let $\underline{v} = (v_{00}, v_{01}, v_{10}, v_{02}, v_{11}, v_{20}, \ldots)$ be the limiting vector containing the steady state probabilities. These probabilities all exist if the arrival rate λ is less than the total

---

[9] We do not claim originality of the block form, but we discovered it independently and the form is useful in that it indicates some nice mathematical properties of the problem.

service rate $\mu_1 + \mu_2$ and if $\mu_1, \mu_2 \neq 0$. We will assume henceforth that the limiting probability vector $\underline{v}$ exists[10].

The vector $\underline{v}$ satisfies the equation $\underline{0} = \underline{v}.\Lambda$. Thus each column of the matrix gives an equation involving some limiting probabilities. If these equations are carefully selected and grouped we can obtain some interesting results about the limiting probabilities. In particular, we need to compare the magnitude of first few components of $\underline{v}$ with the later components for the next chapter.

Property 3.1.[11] For arrival rate λ and service rates μ_1 and μ_2, for $n \geq 1$

$$\lambda \sum_{i+j=n} v_{i,j} = \mu_2 v_{0,n+1} + (\mu_1 + \mu_2) \sum_{i=1}^{n} v_{i,n+1-i} + \mu_1 v_{n+1,0} \quad .$$

Proof: Sum all equations corresponding to all the columns of Λ from (0,0) to (n,0). The result follows. ■

Another way to prove the property is to construct a boundary between the set of states $\{(i, j) \mid i + j \leq n\}$ and the set $\{(i, j) \mid i + j > n\}$ for a fixed n and balance the flow across the boundary.

Moreover, the property can be extended to $n \geq 0$, if $\sum_{i=1}^{0} v_{i,n+1-i}$ is defined to be zero.

Corollary 3.1.1. For arrival rate λ and service rates μ_1 and μ_2, for $n \geq 1$,

$$\sum_{i+j=n+1} v_{i,j} \leq \frac{\lambda}{\min(\mu_1, \mu_2)} \sum_{i+j=n} v_{i,j} \quad ,$$

where $i, j \geq 0$.

---

[10] The existence problem was studied by Kingman (1961). From the state transition rate diagram, we can see that the process is irreducible. Intuitively, $\underline{v}$ is not a zero vector, therefore the process is positive recurrent.

[11] The same result can be found in [Haight58] for the asymmetric JSQ, while our result is for symmetric JSQ.

Proof: From property 3.1., we have

$$\lambda \sum_{i+j=n} v_{ij} = \mu_2 v_{0,n+1} + \mu_1 v_{n+1,0} + (\mu_1 + \mu_2) \sum_{i=1}^{n} v_{i,n+1-i}$$

$$\geq \min(\mu_1, \mu_2) \left\{ v_{0,n+1} + 2 \sum_{i=1}^{n} v_{i,n+1-i} + v_{n+1,0} \right\}$$

$$\geq \min(\mu_1, \mu_2) \sum_{i=0}^{n+1} v_{i,n+1-i} \quad . \quad \blacksquare$$

Corollary 3.1.2. Let λ be the arrival rate, and μ_1, μ_2 be the service rates. Assume n $\geq$ 1, μ_1, $\mu_2 \neq 0$, and $\lambda < \min(\mu_1, \mu_2)$. Then

$$\sum_{i+j \geq n+1} v_{ij} \leq \frac{\lambda}{\min(\mu_1, \mu_2)} \left(1 - \frac{\lambda}{\min(\mu_1, \mu_2)} \right)^{-1} \sum_{i+j=n} v_{ij} \quad .$$

Proof: From Corollary 3.1.1, we have

$$\sum_{i+j \geq n+1} v_{ij} = \sum_{i+j=n+1} v_{ij} + \sum_{i+j=n+2} v_{ij} + \cdots$$

$$\leq \left\{ \frac{\lambda}{\min(\mu_1, \mu_2)} + \left(\frac{\lambda}{\min(\mu_1, \mu_2)} \right)^2 + \cdots \right\} \sum_{i+j=n} v_{ij} \quad .$$

The result follows. $\blacksquare$

Corollary 3.1.3. Let the arrive rate λ be less than the total service rate $\mu_1 + \mu_2$ and assume μ_1, $\mu_2 \neq 0$. If $\mu_1 = \mu_2 = \mu$, then

$$P(\textit{both lines are nonempty}) = \frac{\lambda - \mu}{\mu} + v_{00} \quad .$$

Proof: By Property 3.1, if $\mu_1 = \mu_2 = \mu$, then for k = 0, 1, 2, . . . ,

$$\frac{\lambda}{\mu} \left(\sum_{i+j=k} v_{ij} \right) = v_{0,k+1} + 2 \sum_{i=1}^{k} v_{i,k+1-i} + v_{k+1,0}$$

$$= \sum_{i+j=k+1} v_{ij} + \sum_{i=1}^{k} v_{i,k+1-i} \quad . \tag{7}$$

Summing all the equations in (7) from k = 0 to ∞ gives

$$\frac{\lambda}{\mu} = 1 - v_{00} + \sum_{k=0}^{\infty} \sum_{i=1}^{k} v_{i,k+1-i}$$

$$= 1 - v_{00} + P(\text{both lines are nonempty}) \quad .$$

The result follows. ■

Corollary 3.1.1 is important because it indicates that if $\min(\mu_1, \mu_2)$ is large relative to λ, then the probability of having a total of exactly n+1 customers in the two queues is very small compared to the probability of having exactly n customers in the system.

Corollary 3.1.2 indicates that if $\min(\mu_1, \mu_2)$ is large relative to λ, then the probability of having a total of exactly n customers is high relative to the probability that the system is in a state which represents n+1 or more customers in the system.

The result of Corollary 3.1.3 can be found in [Halfin85] with a different proof.[12]

The following results — Property 3.2, Corollary 3.2.1, and Corollary 3.2.2, indicate that the limiting probabilities v_{02} and v_{20} can also be treated as negligible relative to v_{00}, v_{01}, v_{10}, v_{11} for large $\min(\mu_1, \mu_2)$ and fixed λ.

Property 3.2. Let the arrival rate λ be less than the total service rate $\mu_1 + \mu_2$ and assume $\mu_1, \mu_2 \neq 0$. Then

$$v_{02} + v_{20} \leq \frac{\max(\mu_1, \mu_2)}{\lambda + \min(\mu_1, \mu_2)}(v_{03} + v_{12} + v_{21} + v_{30}) \quad .$$

Proof: According to the fourth and the sixth columns of Λ, we have

$$v_{02}(\lambda + \mu_2) = v_{03}\mu_2 + v_{12}\mu_1 \quad \text{and}$$

$$v_{20}(\lambda + \mu_1) = v_{30}\mu_1 + v_{21}\mu_2 \quad .$$

Thus

$$v_{02}(\lambda + \min(\mu_1, \mu_2)) \leq \max(\mu_1, \mu_2)(v_{03} + v_{12}) \quad ,$$

$$v_{20}(\lambda + \min(\mu_1, \mu_2)) \leq \max(\mu_1, \mu_2)(v_{30} + v_{21}) \quad .$$

Summing the above two equations yields the result. ■

---

[12] We discovered this property independently.

Corollary 3.2.1. With the conditions of Property 3.2, we have

$$(v_{02} + v_{20})\left(1 - \frac{\max(\mu_1, \mu_2)}{\lambda + \min(\mu_1, \mu_2)} \frac{\lambda}{\min(\mu_1, \mu_2)}\right) \leq \frac{\max(\mu_1, \mu_2)}{\lambda + \min(\mu_1, \mu_2)} \frac{\lambda}{\min(\mu_1, \mu_2)} v_{11} \quad .$$

Proof: This result follows from applying Property 3.2 followed by Corollary 3.1.1 and solving for $v_{02} + v_{20}$. ■

Corollary 3.2.2. Assume that $\max(\mu_1, \mu_2) < k \min(\mu_1, \mu_2)$ for some constant $k > 1$. Assume that λ is fixed. Then for $\min(\mu_1, \mu_2)$ sufficiently large, $v_{02} + v_{20}$ can be made arbitrarily small relative to v_{11}.

Proof: The result follows by taking the limit of the left and right hand sides of the inequality in Corollary 3.2.1 as $\min(\mu_1, \mu_2)$ tends to ∞. ■

All of the above results are useful in comparing strategies for a particular customer facing a system with two parallel queues. Queueing systems with high service rates relative to the arrival rate are called light traffic systems. Optimality in light traffic has been studied by Reiman (1987) and Reiman (1989). Corollary 3.2.2 is a result belonging to light traffic queueing systems. We now examine some other results about parallel queues which we can obtain from our formulation.

Property 3.3. Let the arrival rate λ be less than the total service rate $\mu_1 + \mu_2$ and assume $\mu_1, \mu_2 \neq 0$. Let L_i = number of customers in queue i (i = 1, 2). Then

$$\frac{\lambda}{2} P(D = 0) + \lambda P(D > 0) = \mu_1 P(L_1 > 0) \quad \text{and}$$
$$\frac{\lambda}{2} P(D = 0) + \lambda P(D < 0) = \mu_2 P(L_2 > 0)$$

where D is defined to be L_2 - L_1.

Proof: We construct boundaries between the set of states $\{(i, j) \mid i \leq k\}$ and the set $\{(i, j) \mid i > k\}$, for k = 0, 1, 2, ... The flows across the boundary must balance. This

gives the following equations:

$$\begin{aligned}
\mu_1 P(L_1 = 1) &= \lambda P(L_1 = 0) - \frac{\lambda}{2} c_{00} \\
\mu_1 P(L_1 = 2) &= \lambda P(L_1 = 1) - \frac{\lambda}{2} c_{11} - \lambda c_{10} \\
\mu_1 P(L_1 = 3) &= \lambda P(L_1 = 2) - \frac{\lambda}{2} c_{22} - \lambda c_{21} - \lambda c_{20} \\
&\vdots \\
\mu_1 P(L_1 = k) &= \lambda P(L_1 = k-1) - \frac{\lambda}{2} c_{k-1,k-1} - \lambda c_{k-1,k-2} - \cdots - \lambda c_{k-1,0} \\
&\vdots
\end{aligned} \tag{14}$$

Summing up the above equations and substituting D = L_2 - L_1, we have

$$\mu_1 P(L_1 > 0) = \lambda - \frac{\lambda}{2} P(D = 0) - \lambda P(D < 0)$$

$$\Rightarrow \mu_1 P(L_1 > 0) = \frac{\lambda}{2} P(D = 0) + \lambda P(D > 0) \quad .$$

The second property can be proved similarly by constructing boundaries between the set of states {(i, j) | j ≤ k} and the set {(i, j) | j > k}, for k = 0, 1, 2, ... ■

We could have obtained the same result by noting that in equilibrium, the flow into and out of each queue must be equal. The flow into queue 1 is equal to $\lambda\{\frac{1}{2}P(D = 0) + P(D > 0)\}$ where the probability inside the braces is the probability that a job is assigned to queue 1. The flow out of the queue 1 is μ_1 P(L_1 > 0).

Corollary 3.3.1. Assume the conditions of Property 3.3. Then

$$\lambda = \mu_1 P(\text{line 1 is nonempty}) + \mu_2 P(\text{line 2 is nonempty}) \quad . \tag{16}$$

Proof: Sum up the two equations in Property 3.3. The result follows. ■

Corollary 3.3.2. For the conditions of Property 3.3, let p_1 = P(line i is nonempty) for i = 1, 2. Then

$$\max\left(\frac{\lambda - \mu_2}{\mu_1}, 0\right) \leq p_1 \leq \min\left(\frac{\lambda}{\mu_1}, 1\right)$$

with the corresponding result for p_2.

Proof: This follows from Corollary 3.3.1 and the fact that $0 \le p_1 \le 1$. ■

For example, if $\lambda = 1$, $\mu_1 = 5$, $\mu_2 = 0.2$, we obtain

$$p_1 = P(\text{line 1 is nonempty}) \in [.16, .2]$$

.

Corollary 3.3.3 Assume the conditions of Property 3.3. If $\mu_1 = \mu_2 = \mu$, then

$$P(\text{line 1 is nonempty}) = P(\text{line 2 is nonempty}) = \frac{\lambda}{2\mu} \quad .$$

Proof: If $\mu_1 = \mu_2 = \mu$, then P(line 1 is nonempty) = P(line 2 is nonempty). By Corollary 3.3.1, the result follows.

Property 3.4. Let the arrival rate λ be less than the total service rate $\mu_1 + \mu_2$, $D = L_2 - L_1$ and assume $\mu_1, \mu_2 \neq 0$. Then

$$(\lambda + \mu_2)P(D = \quad 1) = \left(\frac{\lambda}{2} + \mu_1\right)P(D = 0) - \mu_1 c_{00} \quad .$$

$$(\lambda + \mu_1)P(D = -1) = \left(\frac{\lambda}{2} + \mu_2\right)P(D = 0) - \mu_2 c_{00} \quad .$$

$$(\lambda + \mu_2)P(D = \quad k + 1) = \mu_1 P(D = \quad k) - \mu_1 c_{0k} \quad \text{and}$$

$$(\lambda + \mu_1)P(D = -k - 1) = \mu_2 P(D = -k) - \mu_2 c_{k0} \quad .$$

for $k = 1, 2, 3, \ldots$.

Proof: Assume that the queueing system is in equilibrium.

The first equation is given by balancing the flow between the set of states $\{(i, j) \mid j - i \le 0\}$ and the set $\{(i, j) \mid j - i > 0\}$.

The second one is given by balancing the flow between the set of states $\{(i, j) \mid j - i \le -1\}$ and the set $\{(i, j) \mid j - i > -1\}$.

The third one is given by balancing the flow between the set of states $\{(i, j) \mid j - i \le k\}$ and the set $\{(i, j) \mid j - i > k\}$.

The fourth one is given by balancing the flow between the set of states $\{(i, j) \mid j - i < -k\}$ and the set $\{(i, j) \mid j - i \geq -k\}$. ■

Corollary 3.4.1.[13] If the conditions of Property 3.4 hold, then

$$\mu_1 P(D < 0) + \mu_2 P(D > 0) = \lambda P(D = 0) \quad .$$

Proof: Sum the third equation in Property 3.4 from k = 1 to ∞ and add to the first equation. This gives

$$\begin{aligned}(\lambda + \mu_2)P(D > 0) &= \mu_1 P(D > 0) + \left(\frac{\lambda}{2} + \mu_1\right)P(D = 0) - \mu_1 P(L_1 = 0) \\ &= -\mu_1 P(D < 0) + \frac{\lambda}{2}P(D = 0) + \mu_1 P(L_1 > 0) \quad .\end{aligned} \tag{22}$$

Sum the fourth equation from k = 1 to ∞ and add to the second equation. This gives

$$(\lambda + \mu_1)P(D < 0) = -\mu_2 P(D > 0) + \frac{\lambda}{2}P(D = 0) + \mu_2 P(L_2 > 0) \quad . \tag{23}$$

Adding equations (22) and (23) together gives

$$\begin{aligned}&(\lambda + 2\mu_2)P(D > 0) + (\lambda + 2\mu_1)P(D < 0) \\ &\qquad = \lambda P(D = 0) + \mu_1 P(L_1 > 0) + \mu_2 P(L_2 > 0) \\ \Rightarrow\ &(\lambda + 2\mu_2)P(D > 0) + (\lambda + 2\mu_1)P(D < 0) \\ &\qquad = \lambda P(D = 0) + \lambda \text{ from Corollary 3.3.1 .}\end{aligned}$$

The result follows.

Corollary 3.4.2.[14] Assume the conditions of the Property 3.4. If $\mu_1 = \mu_2 = \mu$, then

$$P(D = 0) = \frac{\mu}{\lambda + \mu} \quad .$$

where $D = L_2 - L_1$.

---

[13] The same result can be found in [Haight58] for asymmetric JSQ, while our result is for symmetric JSQ.

[14] The same result can be found in [Halfin85] with a different proof.

Proof: If $\mu_1 = \mu_2 = \mu$, then, by symmetry, P(D > 0) = P(D < 0). Hence,

$$\mu(P(D > 0) + P(D < 0)) = \lambda P(D = 0)$$

$$\Rightarrow \mu(1 - P(D = 0)) = \lambda P(D = 0)$$

$$\Rightarrow P(D = 0) = \frac{\mu}{\lambda + \mu} \quad . \qquad \blacksquare$$

Property 3.5. Let the arrival rate λ be less than the total service rate $\mu_1 + \mu_2$ and assume $\mu_1, \mu_2 \neq 0$. Let L_i = number of customers in line i for i = 1, 2. Then

a. $\lambda P(\min(L_1, L_2) = k) = \mu_1 P(L_1 = k+1) + \mu_2 P(L_2 = k+1)$,

for k = 0, 1, 2, . . .

b. $\lambda P(\text{both lines are nonempty}) = \mu_1 P(L_1 \geq 2) + \mu_2 P(L_2 \geq 2)$

c. $\max\left(\frac{\lambda P(\text{both lines are nonempty}) - \mu_2}{\mu_1}, 0\right) \leq P(L_1 \geq 2) \leq \min\left(\frac{\lambda P(\text{both lines are nonempty})}{\mu_1}, 1\right)$

Proof: Part (a) follows from the sum of the k+1st equation in equation set (14) and its complementary equation

$$\mu_2 P(L_2 = k+1) = \lambda P(L_2 = k) - \frac{\lambda}{2} r_{k,k} - \lambda r_{k-1,k} - \cdots - \lambda r_{0,k} \quad .$$

By combining the result of "k = 0" in (a) with equation (16), we obtain (b). By using (b) together with the fact that $0 \leq P(L_2 \geq 2) \leq 1$, we obtain (c). ■

3.4 Analysis Approach II: Approximation by General Two Server Queues with Finite Buffers

A second approach to studying the queueing system A is the following. Suppose that there is a single line which breaks into two waiting/service areas just in front of the two servers. Each waiting/service area or buffer has a finite capacity of size r (see Figure 6). Other waiting customers wait outside these waiting/service areas. As we allow $r \to \infty$, this system tends to the system of two parallel queues with infinite capacity. We call the system B(r).

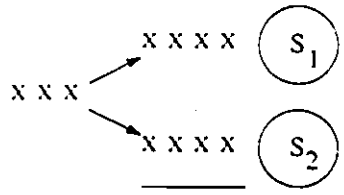

Figure 6 Queueing System B(4)

For r = 1, the states of the system are 0, (0,1), (1,0), 2, 3, ... For r = 2, the states of the system are 0, (0,1), (1,0), (0,2), (1,1), (2,0), 3, 4, ... It is easy to solve for the limiting probabilities $\{v_i\}$ for each of these types of system. For example, if r = 1, the limiting probabilities are:

$$v_0 = \frac{2\mu_1\mu_2(\mu_1+\mu_2-\lambda)}{2\mu_1\mu_2(\mu_1+\mu_2-\lambda)+\lambda(\mu_1+\mu_2)^2}$$

$$v_{01} = \frac{\lambda\mu_1(\mu_1+\mu_2-\lambda)}{2\mu_1\mu_2(\mu_1+\mu_2-\lambda)+\lambda(\mu_1+\mu_2)^2}$$

$$v_{10} = \frac{\lambda\mu_2(\mu_1+\mu_2-\lambda)}{2\mu_1\mu_2(\mu_1+\mu_2-\lambda)+\lambda(\mu_1+\mu_2)^2}$$

$$v_i = \frac{\lambda(\mu_1+\mu_2)(\mu_1+\mu_2-\lambda)}{2\mu_1\mu_2(\mu_1+\mu_2-\lambda)+\lambda(\mu_1+\mu_2)^2}\left(\frac{\lambda}{\mu_1+\mu_2}\right)^{i-1} \qquad \text{for } i = 2,3,\ldots.$$

We observe that the above limiting vectors for r = 1 and i = 0, 2, 3, . . . reduce to the standard results for an M/M/2 queue when $\mu_1 = \mu_2$. This is as expected.

A final comment to this approach — system B(r) should perform exactly the same as the JSQ two parallel queues with r difference jockeying where the customers will switch queues iff the queue length difference is greater than r. Such a system with r difference jockeying was studied by Zhao (1990).

3.5 Analysis Approach III: Approximation by Truncated Capacity Systems

A third approach, similar to the above, examines parallel queues with a finite total capacity.

Suppose we have a parallel system with 2 servers where the total capacity is r = 1. Any customers arriving when the system is full are lost to the system. In this case, the states of the system are (0,0), (0,1), and (1,0). The rate matrix has the same form as the upper left 3×3 corner of Λ in our original Λ except that the diagonal is adjusted so that the rows sum to 0.

Next suppose that we have a parallel system with two servers where the total capacity is r = 2. The rate matrix has the same form as the upper left 6×6 corner of Λ in our earlier large display except that the diagonal is adjusted so that the rows sum to 0. For r = 3 we use the upper left 10×10 corner. We define C(r) to be the sequence of queueing systems labelled with r. As $r \to \infty$, the limiting vector of C(r) approaches the limiting vector of the original system A.

We can solve for the limiting probability vector for each C(r) in this third approach where the Λ is a matrix of finite size. Unfortunately, the results quickly grow in complexity. Using the mathematical computer system MAPLE, we solved for the limiting vector $\underline{v}$ for r = 1, 2, 3, 4, 5, 6. For r = 1, the first component of $\underline{v}$ is

$$v_0 = \frac{2\mu_1\mu_2}{\mu_1 + 2\mu_1\mu_2 + \mu_2} \quad .$$

For r = 2,

$$v_0 = \frac{2\mu_1\mu_2}{1 + \mu_1 + 2\mu_1\mu_2 + \mu_2} \quad .$$

For r = 3, let $A = \mu_1 + 2\mu_1\mu_2 + \mu_2$ and $B = 2\mu_1^2\mu_2 + 2\mu_1\mu_2^2 + 3\mu_1\mu_2 + \mu_1 + \mu_2 + 1$. Then

$$v_0 = \frac{2\mu_2^2\mu_1^2(2\mu_2 + 3 + 2\mu_1)}{AB} \quad .$$

It is interesting to note the position of the zeros in the limiting vectors. Let * denotes any nonzero component.

For r = 1, we have $\underline{v}$ = (*, *, *).

For r = 2, we have $\underline{v}$ = (*, *, *, 0, *, 0).

For r = 3, we have $\underline{v}$ = (*, *, *, *, *, *, 0, *, *, 0).

For r = 2, the zeros correspond to the (0, 2) and (2, 0) components. For r = 3, the zeros correspond to the (0, 3) and (3, 0) components. The zeros occur because, when we truncate the matrix, the states with limiting probability zero are impossible to reach. As r increases, these states are no longer impossible to reach, and a different set of states takes the zero values. This is easy to see with the state transition rate diagrams of these truncated systems C(r). MAPLE suggests that to proceed in this manner without any simplifying assumptions would not be productive.

3.6 Analysis Approach IV: Generating Functions

A fourth approach to studying two parallel queues is to use generating functions. This has been done by Haight (1958), Kingman (1961), and Flatto and McKean (1979). Define

$$G(x,y) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} v_{ij}x^i y^j \quad .$$

Property 3.6. Let $D(x,y) = \sum_{i=0}^{\infty} v_{ii}x^i y^i$, $L(x,y) = \sum_{i<j} v_{ij}x^i y^j$, and $R(x,y) = \sum_{i>j} v_{ij}x^i y^j$. The generating function G(x, y) satisfies the equation

$$\begin{aligned}&(\lambda+\mu_1+\mu_2)G(x,y) - \mu_2 G(x,0) - \mu_1 G(0,y)\\ =&\mu_2\frac{G(x,y)-G(x,0)}{y} + \mu_1\frac{G(x,y)-G(0,y)}{x}\\ &+(x+y)\frac{\lambda}{2}D(x,y) + \lambda(xL(x,y)+yR(x,y)) \quad .\end{aligned} \tag{33}$$

Proof: Consider the original rate matrix A. Each column gives an equation involving the limiting probabilities $\{v_i\}$. We multiply the first equation by x^0y^0, the second equation by x^0y^1, the third equation by x^1y^0, the fourth equation by x^0y^2, and so on. The resulting

equations are :

$$\lambda c_{00}x^0y^0 = \mu_2 c_{01}x^0y^0 + \mu_1 c_{10}x^0y^0,$$

$$(\mu_2 + \lambda)c_{01}x^0y^1 = \mu_2 c_{02}x^0y^1 + \mu_1 c_{11}x^0y^1 + \frac{\lambda}{2}c_{00}x^0y^1,$$

$$(\mu_1 + \lambda)c_{10}x^1y^0 = \mu_2 c_{11}x^1y^0 + \mu_1 c_{20}x^1y^0 + \frac{\lambda}{2}c_{00}x^1y^0,$$

$$(\mu_2 + \lambda)c_{02}x^0y^2 = \mu_2 c_{03}x^0y^2 + \mu_1 c_{12}x^0y^2,$$

$$(\mu_1 + \mu_2 + \lambda)c_{11}x^1y^1 = \mu_2 c_{12}x^1y^1 + \mu_1 c_{21}x^1y^1 + \lambda c_{01}x^1y^1 + \lambda c_{10}x^1y^1,$$

and so on. Summing these equations gives our result. ■

Corollary 3.6.1. Let L_1 = the number of customers in line 1. Then

$$\lambda P(L_1 = 0) = \mu_1 P(L_1 = 1) + \frac{\lambda}{2}c_{00} \quad .$$

Proof: Set x = 0 and y = 1 in equation (33). ■

This result indicates that the flow into the boundary between the set of states {(i, j) | i = 0} and the set {(i, j) | i > 0) is equal to the flow out of the boundary.

3.7 Findings, Extensions, and Comments

In this chapter, we have examined several methods for studying parallel queues with two servers. For each method, we have obtained some results about limiting probabilities.

In particular, we obtained a number of results which compare the relative magnitudes of the sum of all states with the same total number of customers (Property 3.1, Property 3.2, and corollaries). Applications of these results appear in the next chapter. In addition, we have obtained bounds on some useful probabilities (Corollary 3.3.2, Property 3.5(c)). Further, we have obtained some interesting and valuable mathematical relationships (Property 3.3, Corollary 3.3.1, Corollary 3.3.3, Property 3.4, Corollary 3.4.1, Corollary 3.4.2, Property 3.5(a),(b), and equation set (14)).

We found that there are five important equations. They are:

1. $\lambda P(D > 0) + \frac{\lambda}{2} P(D = 0) = \mu_1 P(L_1 > 0)$,
2. $\lambda P(D < 0) + \frac{\lambda}{2} P(D = 0) = \mu_2 P(L_2 > 0)$,
3. $(\lambda + \mu_2) P(D > 0) = -\mu_1 P(D < 0) + \frac{\lambda}{2} P(D = 0) + \mu_1 P(L_1 > 0)$,
4. $(\lambda + \mu_1) P(D < 0) = -\mu_2 P(D > 0) + \frac{\lambda}{2} P(D = 0) + \mu_2 P(L_2 > 0)$, and
5. $1 = P(D > 0) + P(D < 0) + P(D = 0)$,

where D = L_2 - L_1. The first and the second equations are from Property 3.3 and the third and fourth equations are equations (22) and (23). There are five equations and five variables which are important probabilities. However, the rank of the matrix of this set of linear equations is only 4. We can either

1. find one more equation to make the rank equal to five and then directly find the probabilities or
2. find all possible solutions of the above equation set, put in the constraints that all probabilities are in the range of [0,1], and then find bounds on the probabilities.

The second technique was applied to prove Corollary 3.3.2 The solutions of the set of linear equations are as follows, in terms of a parameter t.

$$P(D > 0) = \frac{\mu_1 - (\lambda + \mu_1)t}{\mu_1 - \mu_2} \quad ,$$

$$P(D = 0) = t \quad ,$$

$$P(D < 0) = \frac{\mu_2 - (\lambda + \mu_2)t}{\mu_2 - \mu_1} \quad ,$$

$$P(L_1 > 0) = \frac{2\lambda\mu_1 - \lambda(\mu_1 + \mu_2 + 2\lambda)t}{2\mu_1(\mu_1 - \mu_2)} \quad \text{and}$$

$$P(L_2 > 0) = \frac{2\lambda\mu_2 - \lambda(\mu_1 + \mu_2 + 2\lambda)t}{2\mu_2(\mu_2 - \mu_1)} \quad ,$$

for $0 \leq t \leq 1$.

The advantage of working with just two queues is that many of the problems of parallel queues can be studied without the burden of excessive complexity. Some results presented in this chapter can be extended to multiple parallel queues.

In a multiple parallel queues of n servers, the states can be described as $(l_1, l_2, \ldots, l_n)$ where l_i is the length of the i^{th} queue. Suppose λ is the average arrival rate and μ_i is the average service rate of the i^{th} queue. When we group all the states with the same total number of customers, the state transition matrix will still be in block form. The reason is that transitions from the states of the group corresponding to a total of k customers can only go to states corresponding to a total of k-1 or k+1 customers.

With the block form, Property 3.1 can be extended naturally for n parallel queues as follows:

$$\lambda \sum_{l_1+l_2+\ldots+l_n=k} v_{(l_1,l_2,\ldots,l_n)} = \sum_{l_1+l_2+\ldots+l_n=k+1} \left(\sum_{i=1}^{n} \delta(l_i)\mu_i \right) v_{(l_1,l_2,\ldots,l_n)} .$$

where $\delta(0) = 0$ and $\delta(j) = 1$ for any $j > 0$.

Furthermore, Property 3.3 can be generalized to

$$\lambda P(\text{a job is scheduled to queue i}) = \mu_i P(L_i > 0) \quad .$$

where L_i is the length of the i^{th} queue. The corollaries of Property 3.1 and 3.3 can also be extended.

Finally, we have illustrated that tools such as MAPLE can be useful in queueing studies. A mathematical computing system can be useful for solving large complex systems of equations. It can also be useful for indicating which analytic approaches are most likely to yield (or not to yield) productive results.

CHAPTER 4 OBSERVING QUEUES BEFORE JOINING

In this chapter, we again consider the queueing system A with a buffer where a smart customer S can delay joining a queue until some arrivals or service departures have been observed. The system is called a pseudo parallel queues. All other customers apply strategy 1. Analytic and simulation techniques are used to find conditions under which the smart customer can lower its expected sojourn time in the system by waiting and observing rather than immediately joining the shortest queue.

4.1 Early Research in Control of Two Parallel Queues

The earliest research that we found, started in 1978. Weber (1978) discusses a situation with several identical servers. He shows that if each customer's service time is a random variable with a non-decreasing hazard rate, then the strategy of assigning customers to the shortest queue will maximize the number of customers who complete service by a certain time.

It is known that it is not always optimal for a customer to join the shortest queue if the customer wishes to minimize its expected sojourn time. Whitt (1986) presents a specific example with a carefully chosen distribution for the service time to illustrate this fact.

In his example, there are two servers. The interarrival times are exponentially distributed. The service time distributions for both servers are identical with mass function

$$f(x) = \begin{cases} 1-c & if\ x = 0 \\ c & if\ x = n \end{cases}$$

Whitt shows that the strategy of joining the longer queueing system rather than the shorter one, would decrease the expected total waiting and service times for an arriving customer!

Larsen and Agrawala (1983) consider a case with two servers, but one queue. The servers in their discussion are heterogeneous rather than homogeneous. There is one slow server and one fast server. The slow server is invoked when the queue length of the fast server reaches a threshold level. The slow server continues to function until it completes service on a customer and the queue length of the fast server is less than the threshold level. The authors discuss optimal threshold levels.

Reiman (1989) considers the same problem for light traffic. He shows that if the arrival rate is sufficiently small, the problem becomes equivalent to that of a finite population of customers.

Rothkopf and Rech (1987) discuss parallel queues each with its own server and compare them to a single queue with several servers. Their nonmathematical, but very perceptive, the paper points out some often overlooked considerations which ought to be taken into account when comparing the two types of system. In particular, they observe that when jockeying between queues is permitted, the choice of which of the two types of system is better depends on the criterion used.

In this chapter, we consider a situation with two parallel queues and two possibly heterogeneous servers. The interarrival times are exponentially distributed with rate λ and the service times for the two servers are exponentially distributed with rates μ_1 and μ_2 respectively.

We assume that all customers, with the possible exception of a single smart customer, behave in the usual manner of joining the shortest queue available. If the two queues are of equal length, the regular customers (i.e. all but the smart customer) pick either server with probability 0.5. With apologies to Rothkopf and Rech, no jockeying is permitted.

The smart customer has the option of waiting as long as it wishes before deciding

which queue to join. In this chapter, one strategy has the smart customer observing the system until a service completion occurs. Then the smart customer makes a decision as to which of the two queues to join. During the smart customer's observation time, new customers may join the system and take their place in the queue ahead of the smart customer. This is the cost to the smart customer of delaying its decision.

We present conditions on the rates λ, μ_1, and μ_2, under which the smart customer can lower its total time in the system by observing the system before joining. We present three strategies and make comparisons under various rates. The techniques used include analytic methods and simulation.

4.2 Definitions of Strategies 1, 2, and 3

We are about to define the strategies. They will be studied only for two parallel queues, but the strategies generalize to multiple parallel queues with an arbitrary number of queues. Let n_i be the length of the i^{th} queue, where i = 1, 2, . . . N and N = number of parallel queues. Let $A = \left\{ i \mid n_i = \min_j \{n_j\} \right\}$ and #(A) = number of elements in the set A.

We define strategy 1 to be the JSQ strategy which is:

> Join queue i with probability[15] $\frac{1}{\#(A)}$ for any $i \in A$.

In the case where the number of parallel queues is two, strategy 1 are that S behaves like a regular customer. i.e. It immediately joins the shortest queue, if there is one. If both queues have the same length, S immediately joins either with probability 0.5 .

Strategy 2 is defined as the follows:

---

[15] The probabilities are equal for all queues with the minimum length. We will call the JSQ strategy with this property to be symmetric JSQ.

If $n_i = 0\ \forall i$, then

1. wait for a customer to arrive and complete service.
2. join the queue of the server with the first completed service.

else join queue i with probability $\frac{1}{\#(A)}$ for any $i \in A$.

In the case where the number of parallel queues is two, strategy 2 can be rewritten as follows. If either queue is nonempty, then S behaves as in strategy 1. If both queues are empty (and no customers are being served), S waits for a customer to arrive and complete service. S then joins the queue of the server with the just completed service.

Strategy 3 is defined as the follows:

If $\min_j \{n_j\} > 0$ and $| n_i - n_j | \leq 1\ \forall i, j$, where $1 \leq i < j \leq N$ then

1. waits for the first service completion
2. joins the queue in which the service completion takes place.

else join queue i with probability $\frac{1}{\#(A)}$ for any $i \in A$.

Strategy 3 can be described as follows for two parallel queues. Let n_i be the number of customers in queue i, i = 1, 2 . If both of the queues are nonempty, and $| n_1 - n_2 | \leq 1$, then S waits for the first service completion and joins the queue in which the service completion takes place. Otherwise, S behaves as in strategy 1.

4.3 Comparing Strategy 1 and Strategy 2

To justify our claim that the smart customer can lower its expected total time until completion for certain rates, consider strategies 1 and 2.

If $\mu_1 > \lambda \gg \mu_2$, we expect that strategy 2 should be the better strategy. We now prove that this is indeed the case. The notation $\gg$ means "muchs greater than".

Theorem 4.1. Assume exponential interarrival times and service times with parameters λ, μ_1, μ_2 respectively. Let strategies 1 and 2 be as defined above. Let TCSS = Time to Complete Service of the Smart customer (= sojourn time = total of observation time + waiting time + service time). If $\mu_1 > \lambda \gg \mu_2$, then

$$\text{E(TCSS | Strategy 2)} \leq \text{E(TCSS | Strategy 1)}$$

We will refer to (waiting time + service time) as system time.

Proof: The smart customer behaves in the same manner under strategies 1 and 2 if either queue is nonempty. Thus our comparison of TCSS for the two strategies need only consider the case where both queues are initially empty. Assume for the remainder of the proof that customer S initially arrives at a system which has no customers in either queue (and no customers being served).

Under strategy 1, the expected time for S to complete service is

$$E(\text{TCSS} \mid \text{Strategy 1}) = \frac{1}{2}\left(\frac{1}{\mu_1} + \frac{1}{\mu_2}\right) \tag{40}$$

It is easier to find the value of E(TCSS | Strategy 2) with a state transition probability diagram. See figure 7.

- The state 00 indicates that the system is originally empty and S is observing the system.
- The state 10 represents that a regular customer is being served by server 1 and S is observing the system.
- The state 01 represents that a regular customer is being served by server 2 and S is observing the system.

- The state "More than 2 arrivals" represents the case where there are more than 2 successive arrivals and there is no service completion since the arrival of the smart customer. It is, in fact, a collection of states {(i, j) | i ≥ 1, j ≥ 1 and S is observing}. The smart customer is again observing the system.
- The states "Join Queue 1" and "Join Queue 2" indicate that the first service completion occurs in queue 1 and 2 respectively. They also indicate the decision made by the smart customer. They are {(i, j) | i ≥ 0, j ≥ 0, S joined Queue 1} and {(i, j) | i ≥ 0, j ≥ 0, S joined Queue 2}.

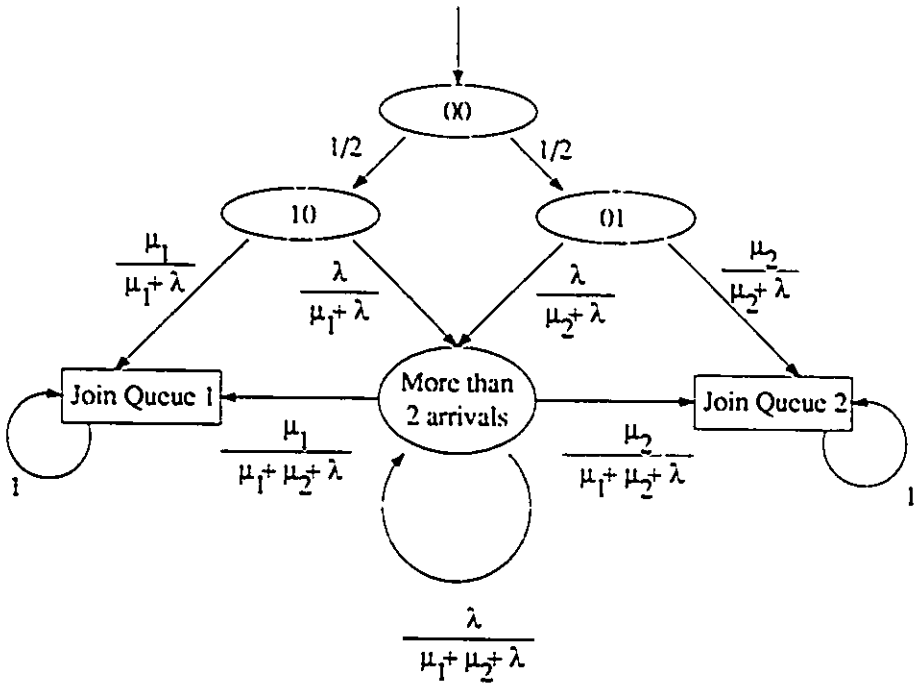

Figure 7 State Transition Probability Diagram for Special Case of Strategy 2

Each edge is labelled with a state transition probability. There is also an expected waiting time associated to each path. Since the arrival process and service completion processes are exponential, it can be shown that the expected waiting time associated with the paths 00 → 01 and 00 → 10 is $1/\lambda$. The expected waiting time for the paths 10 →

"More than 2 arrivals" and 10 → "Join Queue 1" is 1 / (μ_1 + λ). Those for 01 → "More than 2 arrivals" and 01 → "Join Queue 2" is 1 / (μ_2 + λ). For "More than 2 arrivals" to "Join Queue1", "More than 2 arrivals" and "Join Queue 2", the expected waiting times are all 1 / (μ_1 + μ_2 + λ).

Let n denote the number of customers that have arrived before the smart customer makes its decision.

Let (n, i, j) denote the case that n customers arrived before the smart customer decided to join queue j and the first observed customer joined queue i, for n = 1, 2, 3, . . . , i = 1, 2, and j = 1, 2.

Let $p_{n,i,j}$ be the probability that case (n, i, j) occurs.

Let $T_{n,i,j}$ be the expected time to completion of the smart customer in the case (n, i, j).

$p_{n,i,j}$ can be obtained directly by tracing through the path on the state transition diagram and multiplying the probabilities on the path together.

In the case of n = 1, $p_{1,1,1} = \frac{1}{2}\frac{\mu_1}{\mu_1+\lambda}$ and $p_{1,2,2} = \frac{1}{2}\frac{\mu_2}{\mu_2+\lambda}$. There is no such path as 00 → 10 → "Join Queue 2" or 00 → 01 → "Join Queue 1". Therefore, $p_{1,1,2}$ and $p_{1,2,1}$ are both 0.

$$T_{1,1,1} = \frac{1}{\lambda} + \frac{1}{\mu_1+\lambda} + \frac{1}{\mu_1}$$
$$T_{1,2,2} = \frac{1}{\lambda} + \frac{1}{\mu_2+\lambda} + \frac{1}{\mu_2}$$

where their first terms are the expected times waiting for the first arrival, the second terms are the times for the first service completion, and the third terms are the system times of the smart customer.

Let $p_{n,\bullet,\bullet} = \sum_{i=1}^{2}\sum_{j=1}^{2} p_{n,i,j}$.

Accordingly, we have

$$p_{1,\bullet,\bullet}E(\text{TCSS} \mid n = 1) = \frac{1}{2}\left(\frac{\mu_1}{\mu_1+\lambda}\right)\left(\frac{1}{\lambda}+\frac{1}{\mu_1+\lambda}+\frac{1}{\mu_1}\right) + \frac{1}{2}\left(\frac{\mu_2}{\mu_2+\lambda}\right)\left(\frac{1}{\lambda}+\frac{1}{\mu_2+\lambda}+\frac{1}{\mu_2}\right)$$

For n > 1, $p_{n,1,1}$, $p_{n,1,2}$, $p_{n,2,1}$, and $p_{n,2,2}$ can be obtained similarly. They are

$$p_{n,1,1} = \frac{1}{2}\frac{\lambda}{\mu_1+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-2}\frac{\mu_1}{\mu_1+\mu_2+\lambda}$$

$$p_{n,1,2} = \frac{1}{2}\frac{\lambda}{\mu_1+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-2}\frac{\mu_2}{\mu_1+\mu_2+\lambda}$$

$$p_{n,2,1} = \frac{1}{2}\frac{\lambda}{\mu_2+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-2}\frac{\mu_1}{\mu_1+\mu_2+\lambda}$$

$$p_{n,2,2} = \frac{1}{2}\frac{\lambda}{\mu_2+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-2}\frac{\mu_2}{\mu_1+\mu_2+\lambda}$$

The expected waits $T_{n,1,1}$, $T_{n,1,2}$, $T_{n,2,1}$, and $T_{n,2,2}$ are

$$T_{n,1,1} = \frac{1}{\lambda}+\frac{1}{\mu_1+\lambda}+\frac{n-2}{\mu_1+\mu_2+\lambda}+\frac{1}{\mu_1+\mu_2+\lambda}+W_{1,1}$$

$$T_{n,1,2} = \frac{1}{\lambda}+\frac{1}{\mu_1+\lambda}+\frac{n-2}{\mu_1+\mu_2+\lambda}+\frac{1}{\mu_1+\mu_2+\lambda}+W_{1,2}$$

$$T_{n,2,1} = \frac{1}{\lambda}+\frac{1}{\mu_2+\lambda}+\frac{n-2}{\mu_1+\mu_2+\lambda}+\frac{1}{\mu_1+\mu_2+\lambda}+W_{2,1}$$

$$T_{n,2,2} = \frac{1}{\lambda}+\frac{1}{\mu_2+\lambda}+\frac{n-2}{\mu_1+\mu_2+\lambda}+\frac{1}{\mu_1+\mu_2+\lambda}+W_{2,2}$$

where $W_{1,1}$, $W_{1,2}$, $W_{2,1}$, and $W_{2,2}$ represent the expected system time of the smart customer in the different cases. These terms depends on the number of customers lining up in front of the smart customer. If there are k customers, the average system time will be (k+1) / μ_j for the case where the smart customer joined queue j. If n is even, then k = n / 2 - 1 and thus the average system time = $\frac{n}{2\mu_j}$. If n is odd, then k is equal to (n+1) / 2 - 1 or (n-1) / 2 - 1 with equal probability. The reason is that the last customer joined any queue with equal probability in this case. Therefore the average system time is still $\frac{n}{2\mu_j}$.

As a result,

$$W_{1,1} = W_{2,1} = \frac{n}{2\mu_1} \text{ and } W_{1,2} = W_{2,2} = \frac{n}{2\mu_2}$$

Therefore, for n > 1, we have

$$\begin{aligned}
&p_{n,\bullet,\bullet}E(\text{TCSS} \mid \text{ Case n}) \\
&= \frac{1}{2}\left(\frac{\mu_1}{\mu_1+\lambda}\right)\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-1}\left(\frac{1}{\lambda}+\frac{1}{\mu_1+\lambda}+\frac{n-1}{\mu_1+\mu_2+\lambda}+\frac{n}{2\mu_1}\right) \\
&+\frac{1}{2}\left(\frac{\mu_2}{\mu_1+\lambda}\right)\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-1}\left(\frac{1}{\lambda}+\frac{1}{\mu_1+\lambda}+\frac{n-1}{\mu_1+\mu_2+\lambda}+\frac{n}{2\mu_2}\right) \\
&+\frac{1}{2}\left(\frac{\mu_1}{\mu_2+\lambda}\right)\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-1}\left(\frac{1}{\lambda}+\frac{1}{\mu_2+\lambda}+\frac{n-1}{\mu_1+\mu_2+\lambda}+\frac{n}{2\mu_1}\right) \\
&+\frac{1}{2}\left(\frac{\mu_2}{\mu_2+\lambda}\right)\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^{n-1}\left(\frac{1}{\lambda}+\frac{1}{\mu_2+\lambda}+\frac{n-1}{\mu_1+\mu_2+\lambda}+\frac{n}{2\mu_2}\right)
\end{aligned}$$

Consequently,

$$\begin{aligned}
E(\text{TCSS} \mid \textit{Strategy 2}) &= \sum_{n=1}^{\infty} p_{n,\bullet,\bullet}E(\text{TCSS} \mid \text{ Case n}) \\
&= p_{1,\bullet,\bullet}E(\text{TCSS} \mid \text{ Case 1}) + \sum_{n=2}^{\infty} p_{n,\bullet,\bullet}E(\text{TCSS} \mid \text{ Case n}) \\
&= \frac{2\mu_1\mu_2+\lambda\mu_1+\lambda\mu_2}{2\lambda(\mu_1+\lambda)(\mu_2+\lambda)} + \frac{1}{2(\mu_1+\lambda)} + \frac{1}{2(\mu_2+\lambda)} \\
&\quad + \frac{\mu_1+\mu_2+\lambda}{2(\mu_1+\lambda)(\mu_2+\lambda)}\left(2+\frac{\lambda}{\mu_1+\mu_2}+\frac{\lambda(2\mu_1+2\mu_2+\lambda)}{(\mu_1+\mu_2)^2}\right)
\end{aligned} \tag{47}$$

Thus, from (4) and (47). We have for fixed μ_1, λ.

$$\lim_{\mu_2 \to 0} E(\text{TCSS} \mid \textit{Strategy 1}) = \infty \qquad \text{and}$$

$$\lim_{\mu_2 \to 0} E(\text{TCSS} \mid \textit{Strategy 2}) = \frac{4\mu_1^2+5\lambda\mu_1+2\lambda^2}{2\lambda\mu_1^2} < \infty$$

The result follows. ■

Theorem 4.2. Assume exponential interarrival times and service times with parameters as given in Section 1. Let strategies 1 and 2 be as in Theorem 1. Let TCSS = time to complete service of the smart customer (total of observation time plus waiting time plus

service time). Suppose $\mu_1 \gg \lambda > \mu_2$ (and therefore $\mu_1 \to \infty$). Then

$$E(\text{TCSS} \mid Strategy\ 2) \leq E(\text{TCSS} \mid Strategy\ 1) \text{ iff } \lambda > \frac{3+\sqrt{17}}{2}\mu_2$$

Proof: The same proof as used in Theorem 4.1 is still valid until the last paragraph. However, now we have

$$\lim_{\mu_1 \to \infty} E(\text{TCSS} \mid Strategy\ 1) = \frac{1}{2\mu_2} \qquad \text{and}$$

$$\lim_{\mu_1 \to \infty} E(\text{TCSS} \mid Strategy\ 2) = \left(\frac{1}{\lambda} + \frac{\mu_2}{\lambda(\mu_2+\lambda)} + \frac{3}{\mu_2+\lambda}\right)\frac{1}{2}$$

Thus E(TCSS | Strategy 2) ≤ E(TCSS | Strategy 1) iff

$$\left(\frac{1}{\lambda} + \frac{\mu_2}{\lambda(\mu_2+\lambda)} + \frac{3}{\mu_2+\lambda}\right)\frac{1}{2} - \frac{1}{2\mu_2} < 0$$

$$\text{iff} \qquad \frac{-1}{2\mu_2\lambda(\mu_2+\lambda)}\left(\lambda - \frac{3+\sqrt{17}}{2}\mu_2\right)\left(\lambda - \frac{3-\sqrt{17}}{2}\mu_2\right) < 0$$

$$\text{iff} \qquad \lambda > \frac{3+\sqrt{17}}{2}\mu_2 \qquad \blacksquare$$

4.4 Comparing Strategy 1 and Strategy 3

We compare strategy 3 and strategy 1 in this section.

Two lemmas are needed for the comparison of these strategies. The lemma is interesting in its own right.

Lemma 4.1. Assume that queueing system 1 has i customers and queueing system 2 has j customers where | i - j | ≤ 1 and i ≥ 1, j ≥ 1. A new smart customer arrives and waits for the first service completion. This new customer joins the line in which the first completion occurs. The expected sojourn time for the new customer is

$$\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{i+j+1}{\mu_1+\mu_2}.$$

Proof: The proof follows the method used in the proof of theorem 1. Let TC be the Time to Completion of the new customer (or the sojourn time of the new customer.)

Let n be the number of customers that arrive before the first service completion. We group the possible cases for each of n = 0, 1, 2, ... Let (n, j) represent the case where there are n new arrivals before the smart customer joins the queue j. Let $p_{n,j}$ be the probability that the case (n, j) occurs. Let $p_{n,\bullet} = \sum_j p_{n,j}$. The diagram (figure

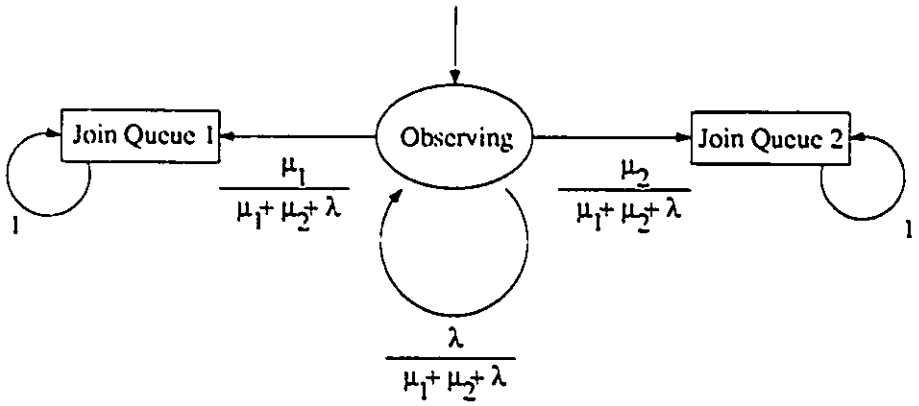

Figure 8 State Transition Probability Diagram for Special Case of Strategy 3

8) is essentially the same as before. Each edge is again labelled with a state transition probability. We omit the states 00, 10, and 01, since they are not included in the set of state $\{(i, j) \mid |\, i - j \,| \le 1, i \ge 1 \text{ and } j \ge 1\}$.

Case 1: Both queueing systems have i customers. By applying the same procedure as in the proof of Theorem 4.1,

$$p_{n,1} = \frac{\mu_1}{\mu_1+\mu_2+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^n, \qquad T_{n,1} = \frac{n}{\mu_1+\mu_2+\lambda} + \frac{1}{\mu_1+\mu_2+\lambda} + \frac{i+\frac{n}{2}}{\mu_1}$$

$$p_{n,2} = \frac{\mu_2}{\mu_1+\mu_2+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^n, \qquad T_{n,2} = \frac{n}{\mu_1+\mu_2+\lambda} + \frac{1}{\mu_1+\mu_2+\lambda} + \frac{i+\frac{n}{2}}{\mu_2}$$

where first terms of $T_{n,1}$ and $T_{n,2}$ are expected waiting times for n arrivals before the first service completion, the second terms are expected waiting times for the first service completion, and the third terms are the expected system times of the smart customer. The procedure used to get the third terms is the same as in the proof of Theorem 4.1. Hence,

$$p_{n,\bullet}E(TC \mid \text{ n arrivals}) = \left((n+1)\frac{\mu_1+\mu_2}{(\mu_1+\mu_2+\lambda)^2} + \frac{2i+n}{\mu_1+\mu_2+\lambda}\right)\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^n$$

and

$$E(TC) = \sum_{n=0}^{\infty} p_{n,\bullet} E(TC \mid \text{ n arrivals})$$

$$= \sum_{n=0}^{\infty} (n+1)\frac{\mu_1+\mu_2}{(\mu_1+\mu_2+\lambda)^2}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^n + \sum_{n=0}^{\infty}\frac{2i+n}{\mu_1+\mu_2+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^n$$

$$= \frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{2i+1}{\mu_1+\mu_2} \quad .$$

Case 2: Queueing system 1 has i + 1 customers and queueing system 2 has i customers, where i ≥ 1.

Using the same method as in Case 1, we obtain

$$E(TC) = \sum_{k=0}^{\infty} (k+1)\frac{\mu_1+\mu_2}{(\mu_1+\mu_2+\lambda)^2}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^k + \sum_{k=0}^{\infty}\frac{2i+1+k}{\mu_1+\mu_2+\lambda}\left(\frac{\lambda}{\mu_1+\mu_2+\lambda}\right)^k$$

$$= \frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{2i+2}{\mu_1+\mu_2} \quad .$$

Combining Case 1 and 2, we obtain

$$E(TC) = \frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{i+j+1}{\mu_1+\mu_2} \quad . \qquad \blacksquare$$

To prove Theorem 4.3 which follows, we need one additional lemma which is stated below. Let $v_{i,j}$ be the limiting probability that the system is in state (i, j) where i is the number of customers in the first queueing system and j is the number of customers in the second queueing system.

Lemma 4.2. Let λ be the arrival rate, and μ_1, μ_2 be the service rates. Assume μ_1, $\mu_2 \neq 0$ and $\lambda < \min(\mu_1, \mu_2)$. Then for k ≥ 2,

$$\sum_{i+j=k} v_{ij} \leq \left(\frac{\lambda}{\min(\mu_1, \mu_2)}\right)^{k-2} \sum_{i+j=2} v_{i,j} \quad .$$

Proof: This follows immediately from Corollary 3.1.1 of *Chapter 3*.

Theorem 4.3. Fix $\lambda > 0$. Assume $0 < a < \mu_1/\mu_2 < b$ for constants a, b. Then for sufficiently large μ_1.

E(TCSS | Strategy 3) < E(TCSS | Strategy 1).

Proof: Let $E_{ij}^{(k)}$ be the expected sojourn time for a new customer which encounters state (i, j) upon entering the system and is using strategy k, k = 1 or 3. We have

$$E(\text{TCSS} \mid Strategy\ 1) = \sum_{i,j} v_{ij} E_{ij}^{(1)} \text{ and}$$
$$E(\text{TCSS} \mid Strategy\ 3) = \sum_{i,j} v_{ij} E_{ij}^{(3)} \quad .$$

Customers act the same under strategies 1 and 3 if

1. $|i - j| > i$ or

2. $|i - j| \leq 1$ and one of i or j is 0,

Let $S = \{(i, j) \mid i > 0, j > 0, |i - j| \leq 1\}$.

Let $E^{(k)} = \sum_{(i,j) \in S} v_{ij} E_{ij}^{(k)}$, k = 1, 3.

Then

$$E(\text{TCSS} \mid Strategy\ 3) - E(\text{TCSS} \mid Strategy\ 1)$$
$$= \sum_{i,j} v_{ij} E_{ij}^{(3)} - \sum_{i,j} v_{ij} E_{ij}^{(1)}$$
$$= \sum_{(i,j) \in S} v_{ij} E_{ij}^{(3)} - \sum_{(i,j) \in S} v_{ij} E_{ij}^{(1)}$$
$$= E^{(3)} - E^{(1)} \quad .$$

Let $P^{(3)} = E^{(3)} - v_{11} E_{11}^{(3)}$.

Then

$$
\begin{aligned}
P^{(3)} &= \sum_{(i,j)\in S} v_{ij} E^{(3)} - v_{11} E_{11}^{(3)} \\
&= \sum_{k=2}^{\infty} \left(v_{k-1,k} E_{k-1,k}^{(3)} + v_{k,k-1} E_{k,k-1}^{(3)} \right) + \sum_{k=2}^{\infty} v_{kk} E_{kk}^{(3)} \\
&= \sum_{k=2}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{2k}{\mu_1+\mu_2} \right) (v_{k-1,k} + v_{k,k-1}) \\
&\quad + \sum_{k=2}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{2k+1}{\mu_1+\mu_2} \right) v_{kk} \qquad \text{(by lemma 4.1)} \\
&\le \sum_{k=2}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{2k}{\mu_1+\mu_2} \right) \left(\sum_{i+j=2k-1} v_{ij} \right) \\
&\quad + \sum_{k=2}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{2k+1}{\mu_1+\mu_2} \right) \left(\sum_{i+j=2k} v_{ij} \right) \\
&= \sum_{k=3}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{k+1}{\mu_1+\mu_2} \right) \left(\sum_{i+j=k} v_{ij} \right) \\
&\le \sum_{k=3}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{k+1}{\mu_1+\mu_2} \right) \left(\frac{\lambda}{\min(\mu_1,\mu_2)} \right)^{k-2} \left(\sum_{i+j=2} v_{ij} \right) \\
&\quad \text{(by lemma 4.2)} \\
&= K_1(\lambda,\mu_1,\mu_2)(v_{02} + v_{11} + v_{20}) \quad .
\end{aligned} \tag{61}
$$

where

$$
K_1(\lambda,\mu_1,\mu_2) = \sum_{k=3}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{k+1}{\mu_1+\mu_2} \right) \left(\frac{\lambda}{\min(\mu_1,\mu_2)} \right)^{k-2} \quad .
$$

Let $L = \frac{\lambda}{\min(\mu_1,\mu_2)}$.

Then

$$
\begin{aligned}
K_1(\lambda,\mu_1,\mu_2) &= \sum_{k=3}^{\infty} \left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{k+1}{\mu_1+\mu_2} \right) L^{k-2} \\
&= \frac{\lambda}{(\mu_1+\mu_2)^2} \frac{L}{1-L} + \frac{L}{\mu_1+\mu_2} \left(\frac{1}{(1-L)^2} + \frac{3}{1-L} \right) \quad .
\end{aligned}
$$

Let $P^{(1)} = E^{(1)} - v_{11}E_{11}^{(1)}$.

Then

$$
\begin{aligned}
P^{(1)} &= \sum_{(i,j)\in S} v_{ij}E_{ij}^{(1)} - v_{11}E_{11}^{(1)} \\
&= \sum_{k=2}^{\infty}\left(v_{k-1,k}E_{k-1,k}^{(1)} + v_{k,k-1}E_{k,k-1}^{(1)}\right) + \sum_{k=2}^{\infty} v_{kk}E_{kk}^{(1)} \\
&= \sum_{k=2}^{\infty}\left(v_{k-1,k}\frac{k}{\mu_1} + v_{k,k-1}\frac{k}{\mu_2}\right) + \sum_{k=2}^{\infty} v_{kk}\left(\frac{k+1}{2\mu_1} + \frac{k+1}{2\mu_2}\right) \\
&\quad \text{(since we are using Strategy 1)} \\
&\le \sum_{k=2}^{\infty}\left(\frac{k}{\mu_1} + \frac{k}{\mu_2}\right)\left(\sum_{i+j=2k-1} v_{ij}\right) + \sum_{k=2}^{\infty}\left(\frac{k+1}{2\mu_1} + \frac{k+1}{2\mu_2}\right)\left(\sum_{i+j=2k} v_{ij}\right) \\
&\le \sum_{k=2}^{\infty}\left(\frac{1}{\mu_1} + \frac{1}{\mu_2}\right)\left(\frac{2k-1}{2} + 1\right)\left(\sum_{i+j=2k-1} v_{ij}\right) \qquad (64) \\
&\quad + \sum_{k=2}^{\infty}\left(\frac{1}{\mu_1} + \frac{1}{\mu_2}\right)\left(\frac{2k}{2} + 1\right)\left(\sum_{i+j=2k} v_{ij}\right) \\
&= \left(\frac{1}{\mu_1} + \frac{1}{\mu_2}\right)\sum_{k=3}^{\infty}\left(\frac{k}{2} + 1\right)\sum_{i+j=k} v_{ij} \\
&\le \left(\frac{1}{\mu_1} + \frac{1}{\mu_2}\right)\sum_{k=3}^{\infty}\left(\frac{k}{2} + 1\right)L^{k-2}\sum_{i+j=2} v_{ij} \qquad \text{(by lemma 4.2)} \\
&= \left(\frac{1}{\mu_1} + \frac{1}{\mu_2}\right)L\left\{\frac{1}{2}(1-L)^{-2} + 2(1-L)^{-1}\right\}(v_{02} + v_{11} + v_{20}) \quad .
\end{aligned}
$$

Let $K_2(\lambda, \mu_1, \mu_2) = \frac{\max(\mu_1,\mu_2)}{\lambda+\min(\mu_1,\mu_2)}L\left(1 - \frac{\max(\mu_1,\mu_2)}{\lambda+\min(\mu_1,\mu_2)}L\right)^{-1}$. By Corollary 3.2.1 of *Chapter 3*, $v_{02} + v_{20} \le K_2(\lambda, \mu_1, \mu_2)\, v_{11}$. So $v_{02} + v_{11} + v_{20} \le (1 + K_2(\lambda, \mu_1, \mu_2))\, v_{11}$.

According to (61) and (64) respectively, we have

$$P^{(3)} \le K_1(\lambda, \mu_1, \mu_2)(1 + K_2(\lambda, \mu_1, \mu_2))v_{11}$$

and

$$P^{(1)} \le K_3(\lambda, \mu_1, \mu_2)(1 + K_2(\lambda, \mu_1, \mu_2))v_{11} \quad .$$

where

$$K_3(\lambda, \mu_1, \mu_2) = \left(\frac{1}{\mu_1} + \frac{1}{\mu_2}\right)L\left\{\frac{1}{2}(1-L)^{-2} + 2(1-L)^{-1}\right\} \quad .$$

So

$$E^{(3)} - E^{(1)} = P^{(3)} - P^{(1)} + c_{11}E_{11}^{(3)} - c_{11}E_{11}^{(1)}$$

$$= c_{11}\left(E_{11}^{(3)} - E_{11}^{(1)}\right) + \left(P^{(3)} - P^{(1)}\right)$$

$$= c_{11}\left(\frac{\lambda}{(\mu_1+\mu_2)^2} + \frac{3}{\mu_1+\mu_2} - \frac{1}{\mu_1} - \frac{1}{\mu_2}\right) + \left(P^{(3)} - P^{(1)}\right) \tag{68}$$

(by lemma 4.1) .

Now

$$\left|P^{(3)} - P^{(1)}\right| \le P^{(3)} + P^{(1)} \qquad \text{since } P^{(3)} \ge 0, P^{(1)} \ge 0$$

$$\le \{K_1(\lambda,\mu_1,\mu_2) + K_3(\lambda,\mu_1,\mu_2)\}(1 + K_2(\lambda,\mu_1,\mu_2))c_{11} \quad .$$

But

$$\frac{3}{\mu_1+\mu_2} - \frac{1}{\mu_1} - \frac{1}{\mu_2} = \frac{-\mu_1\mu_2 - (\mu_1-\mu_2)^2}{(\mu_1+\mu_2)\mu_1\mu_2} < 0 \quad . \tag{70}$$

Let $K_4(\mu_1,\mu_2) = \frac{3}{\mu_1+\mu_2} - \frac{1}{\mu_1} - \frac{1}{\mu_2}$. We wish to show that

$$\lim_{\mu_1\to-\infty} \frac{\lambda/(\mu_1+\mu_2)^2}{|K_4(\mu_1,\mu_2)|} = 0,$$

$$\lim_{\mu_1\to-\infty} \frac{K_1(\lambda,\mu_1,\mu_2)}{|K_4(\mu_1,\mu_2)|} = 0, \text{ and}$$

$$\lim_{\mu_1\to-\infty} \frac{K_3(\lambda,\mu_1,\mu_2)}{|K_4(\mu_1,\mu_2)|} = 0.$$

First

$$\lim_{\mu_1\to-\infty} \frac{\lambda/(\mu_1+\mu_2)^2}{|K_4(\mu_1,\mu_2)|} = \lim_{\mu_1\to-\infty} \frac{\lambda}{(\mu_1+\mu_2)^2}\left(\frac{(\mu_1+\mu_2)\mu_1\mu_2}{\mu_1\mu_2 + (\mu_1-\mu_2)^2}\right)$$

$$= 0 \text{ since } 0 < a < \frac{\mu_1}{\mu_2} < b \quad .$$

Next

$$\lim_{\mu_1\to-\infty} \frac{K_1(\lambda,\mu_1,\mu_2)}{|K_4(\mu_1,\mu_2)|}$$

$$= \lim_{\mu_1\to-\infty} \frac{\mu_1\mu_2}{\mu_1\mu_2 + (\mu_1-\mu_2)^2} \frac{\lambda}{\min(\mu_1,\mu_2) - \lambda}\left\{\frac{\lambda}{\mu_1+\mu_2} + \frac{\min(\mu_1,\mu_2)}{\min(\mu_1,\mu_2)-\lambda} + 3\right\}$$

$$= 0 \text{ since } 0 < a < \frac{\mu_1}{\mu_2} < b.$$

Finally,

$$\lim_{\mu_1\to-\infty} \frac{K_3(\lambda,\mu_1,\mu_2)}{|K_4(\mu_1,\mu_2)|}$$

$$= \lim_{\mu_1\to-\infty} \frac{(\mu_1+\mu_2)^2\lambda}{2(\min(\mu_1,\mu_2,) - \lambda)}\left(\frac{\min(\mu_1,\mu_2)}{\min(\mu_1,\mu_2)-\lambda} + 4\right)\left(\frac{1}{\mu_1\mu_2 + (\mu_1-\mu_2)^2}\right)$$

$$= 0 \text{ since } 0 < a < \frac{\mu_1}{\mu_2} < b \quad .$$

Also, as $\mu_1 \to \infty$,

$$\frac{K_2(\lambda, \mu_1, \mu_2)}{|K_4(\mu_1, \mu_2)|} = \frac{\lambda \max(\mu_1, \mu_2)}{(\lambda + \min(\mu_1, \mu_2)) \min(\mu_1, \mu_2) - \lambda \max(\mu_1, \mu_2)} \frac{(\mu_1 + \mu_2)\mu_1\mu_2}{\mu_1\mu_2 + (\mu_1 - \mu_2)^2}$$

is bounded. From the above limits, we conclude that $\left(\frac{3}{\mu_1+\mu_2} - \frac{1}{\mu_1} - \frac{1}{\mu_2}\right) c_{11}$ dominates the other terms of $E^{(3)}$ - $E^{(1)}$ in (68) as $\mu_1 \to \infty$. From (70), we conclude that $E^{(3)}$ - $E^{(1)} < 0$ for sufficiently large μ_1. The result follows. ■

4.5 Simulation

We will compare our 3 strategies using simulation for various values of λ, μ_1, and μ_2. Without loss of generality, we can take $\lambda = 1$ and vary μ_1, and μ_2. In order that the queueing system have steady state probabilities, we must have the arrival rate less than the total service rate, and both of the service rates greater than zero. Thus we have the conditions that $\mu_1 + \mu_2 > 1$ and $\mu_1, \mu_2 > 0$.

The simulation was written in PASCAL and was carried out on a 386 type microcomputer with a math coprocessor. The speed of the machine is 20 MHz. The simulation program generates its own uniform pseudo random numbers using the multiplicative congruential method. The modulus used was $m = 2^{31} - 1$ and the multiplier was $a = 7^5$. The number of smart customers examined was approximately 6000 per run. Running time for each trial of each pair (μ_1, μ_2) was approximately 2 minutes. There were 16 trials per pair. These confirmed the precision of the results. The actual code appears in [Hlynka89].

We make the following observations about what the expected results should be.

From Theorem 1, we know that if one of the service rates is very small (i.e. close to zero) and the other is quite large, strategy 2 should do better than strategy 1 (where "better" means a lower value of E(TCSS)).

If both μ_1 and μ_2 are large (i.e. light traffic case), then the most common situations would have both queues empty or one queue empty and the other with a single customer.

In such circumstances, waiting for an arrival (strategy 2) would generally take longer than the service time. So strategy 1 should outperform strategy 2 in this case.

Again suppose that μ_1 and μ_2 are large. Strategies 1 and 3 work the same when $| n_1 - n_2 | > 1$ and at least one of n_1 or n_2 is zero. In cases when both queues have exactly one customer, it would be desirable for S to wait and join the queue which empties first. It is possible for an arrival to take place first, but since both μ_1 and μ_2 are large (and $\lambda = 1$), such an event has low probability. Thus we expect strategy 3 to outperform strategy 1 and strategy 2 for large μ_1 and μ_2. Theorem 3 confirms this for strategies 1 and 3.

We now present the results of our simulation study for various values of μ_1 and μ_2.

	$\mu_1 = 2.0$ $\mu_2 = 0.2$	$\mu_1 = 3.0$ $\mu_2 = 0.2$	$\mu_1 = 4.0$ $\mu_2 = 0.2$	$\mu_1 = 5.0$ $\mu_2 = 0.2$
Strategy 1	2.173 ± 0.022	1.626 ± 0.019	1.420 ± 0.014	1.282 ± 0.014
Strategy 2	2.159 ± 0.020	1.603 ± 0.014	1.346 ± 0.011	1.189 ± 0.014
Strategy 3	1.387 ± 0.014	0.991 ± 0.015	0.883 ± 0.011	0.823 ± 0.012

	$\mu_1 = 2.0$ $\mu_2 = 1.5$	$\mu_1 = 3.0$ $\mu_2 = 1.5$	$\mu_1 = 4.0$ $\mu_2 = 1.5$	$\mu_1 = 5.0$ $\mu_2 = 1.5$
Strategy 1	0.642 ± 0.003	0.518 ± 0.002	0.456 ± 0.003	0.418 ± 0.003
Strategy 2	1.496 ± 0.004	1.381 ± 0.006	1.313 ± 0.005	1.269 ± 0.005
Strategy 3	0.627 ± 0.003	0.503 ± 0.002	0.441 ± 0.003	0.402 ± 0.002

	$\mu_1 = 4.0$ $\mu_2 = 3.0$	$\mu_1 = 5.0$ $\mu_2 = 3.0$	$\mu_1 = 5.0$ $\mu_2 = 4.0$	$\mu_1 = 5.0$ $\mu_2 = 5.0$
Strategy 1	0.300 ± 0.001	0.270 ± 0.001	0.229 ± 0.0004	0.203 ± 0.001
Strategy 2	1.257 ± 0.005	1.228 ± 0.005	1.207 ± 0.005	1.188 ± 0.004
Strategy 3	0.296 ± 0.001	0.267 ± 0.001	0.227 ± 0.0004	0.202 ± 0.001

Table 3 90% Confidence Intervals for the Means of Simulated Average Sojourn Times for all Smart Customers using Strategies 1, 2, and 3.

	$\mu_1 = 1.0$ $\mu_2 = 1.0$	$\mu_1 = 2.0$ $\mu_2 = 2.0$	$\mu_1 = 3.0$ $\mu_2 = 3.0$	$\mu_1 = 4.0$ $\mu_2 = 4.0$
Strategy 1	1.343 ± 0.006	0.541 ± 0.003	0.347 ± 0.001	0.256 ± 0.001
Strategy 2	2.012 ± 0.007	1.434 ± 0.004	1.301 ± 0.005	1.230 ± 0.005
Strategy 3	1.318 ± 0.007	0.531 ± 0.002	0.343 ± 0.001	0.254 ± 0.001

	$\mu_1 = 0.6$ $\mu_2 = 0.6$	$\mu_1 = 0.7$ $\mu_2 = 0.4$
Strategy 1	4.580 ± 0.049	7.868 ± 0.137
Strategy 2	4.934 ± 0.049	8.199 ± 0.118
Strategy 3	4.686 ± 0.029	7.818 ± 0.084

We observe that strategy 2 beats strategy 1 only in the cases when $\mu_2 = .2$ and $\mu_1 = 2, 3, 4, 5$.

We observe that strategy 3 beats both strategy 1 and strategy 2 in every case chosen for the simulation study except when $\mu_1 = 0.6$ and $\mu_2 = 0.6$. This remarkable performance of strategy 3 versus strategy 1 even in cases where μ_1 and μ_2 were equal and small was a surprise. This suggests that strategy 3 is generally a better strategy than strategy 1.

4.6 Findings

The objective of the chapter was to show mathematically that currently used strategies of immediately joining the shortest queue (strategy 1) are not always optimal. We have presented 2 alternative strategies, both of which involved observing the system before joining. We worked with two parallel queues and assumed exponential interarrival times and exponential service times. We proved that for certain parameter values, the 2 new strategies will outperform strategy 1. Our simulation studies indicate that strategy 3 is better than strategy 1 for most parameter values that we would expect to encounter in actual practice.

The simulation showed that μ_1 and μ_2 need not be large in order to have strategy 3 outperform strategy 1.

CHAPTER 5 CONCLUSIONS

5.1 Relationship with Other Work

Control of multiple server queues has been studied for more than three decades although control of parallel queues has been studied for a shorter period of time. A detailed survey appears in Appendix A of this thesis. The studies can be classified into

1. optimizing individual objectives and
2. optimizing social objectives.

The research in optimizing individual objectives is still young. Almost all published papers are concerned with the social objectives. However, our focus is minimizing sojourn time of smart customers (an individual objective) for the parallel queues. Studies of this area do not seem to exist.

5.2 An Interesting Problem

In the study of the properties of queueing system A, we found an interesting problem.

From Corollary 3.3.3, if $\mu_1 = \mu_2 = \mu$, then P(line 1 in nonempty) = P(line 2 in nonempty) = $\lambda/2\mu$.

We attempted to consider each of the queues in queueing system A as a single queueing system. Thus each system has its own arrival process, its own server and its own queue. Since we had an unknown arrival process and an exponential server, we naturally attempted to model this subsystem as a G/M/1 queueing system. With this model, we could find the queue length probability distribution of the subsystem.

Unfortunately, we finally rejected the argument. The reason was that the arrival process depends not only on the current state of the system, but also on the current state of another "invisible" system. The problem was discarding some important information.

Usually some statistical distribution is assumed for the unknown arrival processes, when some real world systems (e.g. restaurants) are modelled. This problem suggests that such assumptions could be dangerous, if we do not take a macroscopic point of view of the system and proceed to model the arrival process with a general probability distribution.

5.3 Summary of Findings

1. The importance of parallel queueing models was demonstrated.
2. Many mathematical properties of the queueing system A were discovered. They include Properties 3.2, 3.3, 3.4, 3.5, 3.6. Property 3.1 is similar to a result in [Haight58], but the focus of Haight's work was an asymmetric JSQ with probability 1.0 of assigning to the first queue in the case of equal queue length. We concentrated on symmetric JSQ. We discovered Property 3.1, Corollary 3.4.1, and Property 3.3 independently.
3. Corollaries 3.1.1, 3.1.2, 3.2.1, and 3.2.2 were established for comparing strategies 1 and 3. Corollaries 3.1.3, 3.3.1, 3.3.2, 3.3.3, 3.4.1, 3.4.2 and 3.6.1 are elegant and interesting properties. Corollary 3.4.1 is again similar to that in [Haight58]. Corollary 3.4.2 can be found in [Halfin85] with different proof.
4. By setting up boundaries between sets of states, we obtained two properties (3.1 and 3.4) and one set of equations (14).
5. We attempted to discover more properties of the queueing system A by studying two similar queueing systems B(r) and C(r). We also mentioned the similarity between the queueing system B(r) and Zhao's parallel queues with r difference jockeying [Zhao90]. We found that Approaches II and III were not very successful.

6. The generating function approach[16] is still productive for our problem and Corollary 3.6.1 was proved with this approach.
7. We grouped five relations among five important probabilities $P(D = 0)$, $P(D > 0)$, $P(D < 0)$, $P(L_1 > 0)$, and $P(L_2 > 0)$ in Section 3.7.
8. It is possible to extend the properties obtained in Chapter 3 to multiple parallel queues.
9. A technique was proposed to obtain the bounds on some probabilities. It is applied in Corollary 3.3.2. It can also be applied to find bounds for $P(D = 0)$, $P(D > 0)$, $P(D < 0)$, $P(L_1 > 0)$, and $P(L_2 > 0)$ with the relations presented in Section 3.7.
10. We proved that strategy 2 performs better than strategy 1 under the condition that $\mu_1 > \lambda \gg \mu_2$ (i.e. $\mu_2 \to 0$) and $\mu_1 \gg \lambda > \frac{3+\sqrt{17}}{2}\mu_2$.
11. We proved that strategy 3 performs better than strategy 1 under the condition that μ_1 and μ_2 are large (and of the same order.)
12. Through simulation, we found that strategy 3 performs the best among three strategies even for "not very large μ_1 and μ_2."

5.4 Conclusions of the Thesis

The study showed that strategy 2 and strategy 3 do perform better than strategy 1 in some cases. As mentioned in Chapter 2, strategy 2 and strategy 3 are designed based on the heuristic that "gathering information is the first rule of winners." Therefore, we showed the thesis statement — *under certain circumstances, the average sojourn time of an individual job can be shortened by gathering information about the pseudo parallel queues before joining the queue.*

Perhaps, this result could help us define the feasible solution space, and hence find the optimal solution.

---

[16] The approach was first applied by Haight in [Haight58].

5.5 Future Work

First, the queueing model can be extended in the thesis. The current state of the queueing system A is assumed to be known to the customers all the time. This may not be valid when modelling distributed systems with slow communication links. The author attempted to extend the model with the following enhancement:

1. The current state of the whole system is unknown to all customers.
2. The current state of a queue is only known to the customers that are in the queue.
3. All customers get information about the state of the system (newspaper) from messages (mail). The average time between messages being sent and received is equal to $1/\alpha$, where $\alpha > 0$. In other words, the newspaper is, on average, $1/\alpha$ units of time old.
4. The average delay time for a customer to join a queue from the shared buffer or to switch queues is equal to $1/\beta$, where $\beta > 0$.

The second assumption may not be valid for some systems, but it is true for a distributed system. Figure 9 indicates a typical realization of this model.

The modified queueing model was studied for a limited time. We found that the next state of the system depends on its entire history, not just the current state. Therefore, the system is not a Markov process.

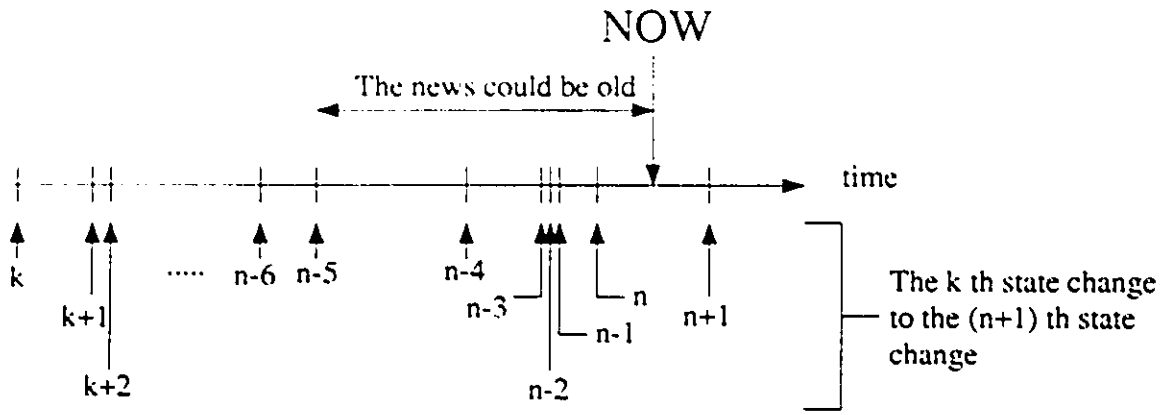

Figure 9 The Next State Depends the System's Entire History.

A second direction of future work is as follows. We may be able to find one more relation among $P(L_1 > 0)$, $P(L_2 > 0)$, $P(L_1 > L_2)$, $P(L_1 < L_2)$, and $P(L_1 = L_2)$. If so, we can solve for these important probabilities.

The third direction of new work would be to propose and study new join-queue strategies. The ultimate objective is to find the optimal solution to the problem.

BIBLIOGRAPHY

Adan, I. J. B. F., Wessels, J., and Zijm, W. H. M. Analysis of the asymmetric shortest queue problem. *Queueing Systems Theory and Applications 8* (1991), 1–58.

Agrawala, A. K., Coffman, E. G., Garey, M. R., and Tripathi, S. K. A stochastic optimization algorithm minimizing expected flow times on uniform processors. *IEEE Transactions on Computers 33* (1984), 351–356.

Baccelli, F., and Makowski, A. Simple computable bounds for the fork-join queue. In *Proc. John Hopkins Conf. Inf. Sci.* (1985), John Hopkins University, pp. 436–441.

Bell, C. E. Turning off a server with customers present: Is this any way to run an M/M/c queue with removable servers? *Operations Research 23* (1975), 517–574.

Bell, C. E. Optimal operation of an M/M/2 queue with removable servers. *Operations Research 28* (1980), 1189–1204.

Bell, C. E., and Stidham, S. Individual versus social optimization in the allocation of customers to alternative servers. *Management Sciences 29* (1983), 831–839.

Benes, V. *Mathematical Theory of Connecting Networks and Telephone Traffic.* Academic Press, New York, 1965.

Blanc, J. P. C. Asymptotic analysis of a queueing system with a two-dimensional state space. *Journal of Applied Probability 21* (1984), 870–886.

Blanc, J. P. C. A note on waiting times in systems with queues in parallel. *Journal of Applied Probability 24* (1987), 540–546.

Boel, R. K., and Schuppen, J. H. Distributed routing for load balancing. In *Proceeding of the IEEE* (1989), vol. 77, pp. 210–221.

Bonomi, F., and Kumar, A. Adaptive optimal load balancing in a nonhomogeneous multiserver system with a central job scheduler. *IEEE Transactions on Computers 39* (1990), 1232–1250.

Brandwajn, A. An iterative solution of two-dimensional birth and death processes. *Operations Research 27* (1979), 595–603.

Brumelle, S. L. Some inequalities for parallel-server queues. *Operations Research 19* (1971), 402–413.

Chang, C.-J., and Chang, J.-F. The effect of routing and buffer sharing on the behavior of a finite queue with batch Poisson inputs and synchronous servers. *SIAM Journal of Applied Mathematics 47* (1987), 352–366.

Chow, Y. C., and Kohler, W. H. Models for dynamic load balancing in a heterogeneous multiple processor system. *IEEE Trans. Comput. 28* (1979), 354–361.

Cinlar, E. *Analysis of Systems of Queues in Parallel.* PhD thesis, University of Michigan, 1965.

Cohen, J. W., and Boxma, O. J. *Boundary Value Problems in Queueing System Analysis.* North Holland, 1983.

Conolly, B. W. The autostrada queueing problem. *Journal of Applied Probability 21* (1984), 394–403.

Courcoubetis, C., and Varaiya, P. Optimal resource allocation for two processes. *AT&T Technical Journal 64* (1985), 1–14.

Courcoubetis, C. A., and Reiman, M. I. Optimal control of a queueing system with simultaneous service requirements. *IEEE Transcations on Automatic Control 32* (1987), 717–727.

Courcoubetis, C. A., and Reiman, M. I. Optimal dynamic allocation of heterogeneous servers under the condition of total overload: The discounted case. In *Proc. 27th IEEE Conf. on Decision and Contr.* (1988), pp. 634–639.

Crabill, T. B., Gross, D., and Magazine, M. J. A classified bibliography of research on optimal design and control of queues. *Operations Research 25* (1977), 219–232.

Csiszàr, I. I-divergence geometry of probability distributions and minimzation problems. *Ann. Prob. 3* (1975), 146–158.

Daduna, H. Optimal control of a queueing system with an exponential and an Erlangian server and renewal input stream. *Optimization 18* (1987), 883–892.

Dave, U., and Shah, Y. K. Maximum likelihood estimates in a M/M/2 queue with heterogeneous servers. *The Journal of Operational Research Society 31* (1980), 423–426.

Davis, F. *Optimal Control of Arrivals to a Two-Server Queueing System with Separate Queues*. PhD thesis, North Carolina State University at Raleigh, 1977.

Disney, R. L., and Mitchell, W. E. A solution for queues with instantaneous jockeying and other customer selection rules. *Naval Research Logistics Quarterly 17* (1971), 315–325.

Duda, A., and Czachorski, T. Performance evaluation of fork and join synchronization primitives. *Acta Informatica 24* (1987), 525–553.

Elsayed, E. A., and Bastani, A. General solutions of jockeying problem. *European Journal of Operational Research 22* (1985), 387–396.

Ephremides, A., Varaiya, P., and Walrand, J. A simple dynamic routing problem. *IEEE Transaction on Automatic Control 25* (1980), 690–693.

Fayolle, G., King, P. J. B., and Mitrani, I. The solution of certain two-dimensional Markov models. *Advances in Applied Probability 14* (19[illegible], 295–308.

Ferdinand, A. E. A statistical mechanics approach to system analysis. *IBM Journal of Research and Development 14* (1970), 539–547.

Flatto, L. Two parallel queues created by arrival with two demands II. *SIAM Journal of Applied Mathematics 45* (1985), 861–878.

Flatto, L. The longer queue model. *Prob. in the Eng. and Info. Sci. 3* (1989), 537–559.

Flatto, L., and Hahn, S. Two parallel queues created by arrival with two demands I. *SIAM Journal of Applied Mathematics 44* (1984), 1041–1053.

Flatto, L., and McKean, H. P. Two parallel queues with equal servicing rates. Tech. rep., Science Report of IBM, 1977.

Flatto, L., and McKean, H. P. Two queues in parallel. *Communications on Pure and Applied Mathematics 30* (1977), 255–263.

Foschini, G. *Computer Performance*. Amsterdam, The Netherlands: North-Holland, 1977, ch. on Heavy Traffic Diffusion Analysis and Dynamic Routing in Packet-Switched Networks, pp. 499–513.

Foschini, G., and Salz, J. A basic dynamic routing problem and diffusion. *IEEE Transactions on Communications 26* (1978), 320–327.

Foschini, G. J. Equilibria for diffusion models for pairs of communicating computers-symmetric case. *IEEE Transactions on Information Theory 28* (1982), 273–284.

Gail, H. R., Hantler, S. L., and Taylor, B. A. Analysis of a non-preemptive priority multiserver queue. *Advances in Applied Probability 20* (1988), 852–879.

Gaver, D. P. Diffusion approximations and models for certain congestion problems. *Journal of Applied Probability 5* (1968), 607–623.

Gertsbakh, I. Shorter queue problem: A numerical study using the matrix geometric solution. *European Journal of Operations Research 15* (1984), 374–381.

Giorno, V., Nobile, A. G., and Ricciardi, L. M. On some diffusion approximations to queueing systems. *Advances in Applied Probability 18* (1986), 991–1014.

Giorno, V., Nobile, A. G., and Ricciardi, L. M. On some time-non-homogeneous diffusion approximations to queueing systems. *Advances in Applied Probability 19* (1987), 974–994.

Glazebrook, K. D., and Nash, P. On multi-server stochastic scheduling. *Royal Statistical Society Journal, Series B 38* (1976), 67–72.

Glazer, H. Jockeying in queues. *Operations Research 6* (1958), 145.

Grassmann, W. K. Transient and steady state results for two parallel queues. *Omega 8* (1980), 105–112.

Gubner, J. A., Gopinath, B., and Varadhan, S. R. S. Bounding functions of Markov processes and the shortest queue problem. *Advances in Applied Probability 21* (1989), 842–860.

Guiasu, S. *Information Theory with Applications*. McGraw-Hill, New York, 1977.

Guiasu, S. Maximum entropy condition in queueing theory. *The Journal of Operational Research Society 37* (1986), 293–301.

Gumbel, H. Waiting lines with heterogeneous servers. *Operations Research 8* (1960), 504–511.

Haight, F. A. Two queues in parallel. *Biometrika 45* (1958), 401–410.

Halfin, S. The shortest queue problem. *Journal of Applied Probability 22* (1985), 865–878.

Hall, W. K., and Disney, R. L. Finite queues in parallel under a generalized channel selection rule. *Journal of Applied Probability 8* (1971), 413–416.

Helm, Werner E. aand Waldmann, K.-H. Optimal control of arrivals to multiserver queues in a random environment. *Journal of Applied Probability 21* (1984), 602–615.

Herzog, U., Woo, L., and Chandy, K. M. Solution of queueing problems by a recursive technique. *IBM Journal of Research and Development 19* (1975), 295–300.

Hlynka, M., and Poon, W.-H. A computer simulation program for parallel queues. Tech. rep., University of Windsor, 1989.

Hlynka, M., Stanford, D., Poon, W. H., and Wang, T. Observing queues before joining. to appear Operations Research, 1991.

Howard, R. A. *Dynamic Programming and Markov Processes*. Cambridge, MA: M.I.T. Press, 1960.

Huang, C. C., Brumelle, S. L., Sawaki, K., and Vertinsky, I. Optimal control for multi-server queuing systems under periodic review. *Naval Research Logistics Quarterly 24* (1977), 127–135.

Hwang, K., and Briggs, F. A. *Computer Architecture and Parallel Processing*. McGraw Hill, 1984.

Iglehart, D. L. Limiting diffusion approximations for the many server queue and the repairman problem. *Journal of Applied Probability 2* (1965), 429–441.

Iliadis, I., and Lien, L. Y.-C. Resequencing delay for a queueing system with two heterogeneous servers under a threshold-type scheduling. *IEEE Transactions on Communications 36* (1988), 692–702.

Jaynes, E. T. Information theory and statistical mechanics. *Physical Review 106* (1957), 620–630.

Jones, L. K. Approximation-theoretic derivation of logarithmic entropy principles for inverse problems and unique extension of the maximum entropy method to incorporate prior knowledge. *SIAM Journal of Applied Mathematics 49* (1989), 650–661.

Kao, E. P. C., and Lin, C. A matrix-geometric solution of the jockeying problem. *European Journal of Operational Research 44* (1990), 67–74.

Kella, O., and Whitt, W. Diffusion approximations for queues with server vacations. *Advances in Applied Probability 22* (1990), 706–729.

Kesavan, H. K. On the families of solutions to generalized maximum entropy and minimum cross-entropy problems. *General Systems 16* (1990), 199–214.

Kingman, J. F. C. The single server queue in heavy traffic. In *Proceedings of the Cambridge Philosophical Society* (1961), vol. 57, pp. 902–904.

Kingman, J. F. C. Two similar queues in parallel. *The Annals of Mathematical Statistics 32* (1961), 1314–1323.

Kleinrock, L. *Queueing Systems*, vol. 1. A Wiley-Interscience publication, 1975.

Kleinrock, L. *Queueing Systems*, vol. 2. A Wiley-Interscience publication, 1976.

Knessl, C., Matkowsky, B. J., Schuss, Z., and Tier, C. Two parallel M/G/1 queues where arrivals join the system with the smaller buffer content. *IEEE Transactions on Communications 35* (1987), 1153–1158.

Knessl, C., Matkowsky, B. J., Schuss, Z., and Tier, C. The two repairman problem: A finite source M/G/2 queue. *SIAM Journal of Applied Mathematics 47* (1987), 367–397.

Knessl, C., Matkowsky, B. J., Schuss, Z., and Tier, C. An integral equation approach to the M/G/2 queue. *Operations Research 38* (1990), 506–518.

Knudsen, N. C. Individual and social optimization in a multi-server queue with a general cost-benefit structure. *Econometrica 40* (1972), 515–528.

Koenigsberg, E. On jockeying in queues. *Management Science 12* (1966), 412–436.

Kouvatsos, D. D., and Othman, A. T. Optimal flow control of a G/G/c finite capacity queue. *The Journal of Operational Research Society 40* (1989), 659–670.

Kouvatsos, D. D., and Tabet-Aouel, N. A maximum entropy priority approximation for a stable G/G/1 queue. *Acta Informatica 27* (1989), 247–286.

Krishnamoorthi, B. On Poisson queues with two heterogeneous servers. *Operations Research 11* (1963), 321–330.

Krishnan, K. R. Joining the right queue: A state-dependent decision rule. *IEEE Transactions on Automatic Control 35* (1990), 104–108.

Larsen, R. L. *Control of Multiple Exponential Servers with Application to Computer Systems*. PhD thesis, Dep. Comput. Sci., University of Maryland, 1981.

Larsen, R. L., and Agrawala, A. K. Control of a heterogeneous two server exponential queueing system. *IEEE Transactions on Software Engineering 9* (1983), 522–526.

Lazar, R. L. Optimal flow control of an M/M/m queue. *Journal of the Association for Computing Machinery 31* (1984), 86–98.

Lehtonen, T. *On the Optimality of the Shortest Line Discipline. Chapter 4 in Stochastic Comparisons for Many Server Queues*. PhD thesis, The Helsinki School of Economics, 1981.

Lemoine, A. J. A queuing system with heterogeneous servers and autonomous traffic control. *Operations Research 23* (1975), 681–686.

Lin, W., and Kumar, P. R. *Analysis and Optimization of Systems*. Berlin, West Germany: Springer-Verlag, 1982, ch. Stochastic Control of a Queue with Two Servers of Different Rates, pp. 719–728.

Lin, W., and Kumar, P. R. Optimal control of a queueing system with two heterogeneous servers. *IEEE Transactions on Automatic Control 29* (1984), 696–703.

Lippman, S. A. Applying a new device in the optimization of exponential queueing systems. *Operations Research 23* (1975), 687–710.

Magazine, M. J. Optimal control of multi-channel service systems. *Naval Research Logistics Quarterly 18* (1971), 177–183.

Nelson, R., and Tantawi, A. Approximate analysis of fork/join synchronization in parallel queues. *IEEE Transactions on Computers 37* (1988), 739–743.

Nelson, R., Towsley, D., and Tantawi, A. Performance analysis of parallel processing systems. *IEEE Transactions on Software Engineering 14* (1988), 532–540.

Nelson, R. D., and Philips, T. K. An approximation to the response time for the shortest queue routing. *Performance Evaluation Review 17* (1989), 181–189.

Neuts, M. F. *Matrix-Geometric Solution in Stochastic Models*. Johns Hopkins University Press, Baltimore, 1981.

Ohta, H., and Furukawa, M. An application of the number of minimal lattice paths. *The Journal of Operational Research Society 36* (1985), 1117–1124.

Papoulis, A. *Probability and Statistics*. Prentice Hall, 1990, ch. Entropy, pp. 414–436.

Parthasarathy, P. R., and Sharafali, M. Transient solution to the many-server Poisson queue: A simple approach. *Journal of Applied Probability 26* (1989), 584–594.

Pawlikowski, K. Steady-state simulation of queueing processes: A survey of problems and solutions. *ACM Computing Surveys 22* (1990), 123–170.

Ramachandran, K. M. Nearly optimal control of queues in heavy traffic with heterogeneous servers. *Stochastic Analysis and Applications 7* (1989), 211–234.

Ramaswami, V., and Latouche, G. A general class of Markov processes with explicit matrix-geometric solutions. *OR Spektrum 8* (1986), 209–218.

Rao, B. M., and Posner, M. Algorithmic and approximation analysis of the shorter queue model. *Naval Research Logistics 34* (1987), 381–398.

Reiman, M. I. Asymptotically optimal trunk reservation for large trunk groups. In *Proc. 28th IEEE Conf. on Decision and Control* (1989), pp. 2536–2541.

Reiman, M. I. Optimal control of a heterogeneous two server queue in light traffic. Conference Paper: CORS/TIMS/ORSA Joint National Meeting, 1989.

Reiman, M. I., and Simon, B. Open queueing systems in light traffic. *Mathematics of Operations Research 14* (1989), 26–59.

Rhee, H.-K., and Sivazlian, B. D. Distribution of the busy period in a controllable M/M/2 queue operating under the triadic (O, K, N, M) policy. *Journal of Applied Probability 27* (1990), 425–432.

Rosberg, Z., and Karmani, P. Customer routing to different servers with complete information. *Advances in Applied Probability 21* (1989), 861–882.

Rosberg, Z., and Makowski, A. M. Optimal routing to parallel heterogeneous servers – small arrival rates. *IEEE Transactions on Automatic Control 35* (1990), 789–796.

Rothkopf, M. H., and Rech, P. Perspectives on queues: Combining queues is not always beneficial. *Operations Research 35* (1987), 906–909.

Rubinovitch, M. The slower server problem: a queue with stalling. *Journal of Applied Probability 22* (1985), 879–892.

Rudin, H. On routing and delta-routing: A taxonomy and performance comparison of techniques for packet-switched networks. *IEEE Transaction on Communications 24* (1976), 43–59.

Sato, M., and Cong, T. T. The number of minimal lattice paths restricted by two parallel lines. *Discrete Math. 43* (1983), 249–261.

Sauer, C. H., and Chandy, K. M. *Computer Systems Performance Modelling*. Prentice-Hall, 1981.

Schassburger, R. A service system with two parallel queues. *Journal of Computing* (1967), 24–29.

Shannon, C. E. A mathematical theory of communication. *Bell System Technical Journal 27* (1948), 379–423,623–656.

Sharma, O. P., and Dass, J. Multi-server Markovian queue with finite waiting space. *Sankhya, Series B 50* (1988), 428–431.

Shore, J. E. Information theoretic approximations for M/G/1 and G/G/1 queueing systems. *Acta Infomatica 17* (1982), 43–61.

Shore, J. E., and Johnson, R. W. Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transactions on Information Theory 26* (1980), 26–37.

Stein, R. M. Scaling up: Get the message? *BYTE 16* (1991), 231–240.

Stidham, S. Optimal control of admission to a queueing system. *IEEE Transactions on Automatic Control 30* (1985), 705–713.

Takács, L. *Combinatorial Methods in the Theory of Stochastic Processes*. Wiley (New York), 1967.

Takagi, H., Ed. *Stochastic Analysis of Computer and Communication Systems*. North-Holland, Amsterdam, 1990.

Tanenbaum, A. S., and VanRenesse, R. Distributed operating systems. *ACM Computing Surveys 17* (1985), 419–470.

Towsley, D., Rommel, C. G., and Stankovic, J. A. Analysis of fork-join program response times on multiprocessors. *IEEE Transactions on Parallel and Distributed Systems 1* (1990), 286–303.

Walrand, J. A note on optimal control of a queueing system with two heterogeneous servers. *System & Control Letters 4* (1984), 131–134.

Wang, T., and Hlynka, M. Two parallel queues with partial jockeying. Tech. rep., University of Windsor, 1990.

Wang, Y.-T., and Morris, R. J. T. Load sharing in distributed systems. *IEEE Transactions on Computers 34* (1985), 204–217.

Weber, R. R. On the optimal assignment of customers to parallel servers. *Journal of Applied Probability 15* (1978), 406–413.

Whitt, W. Deciding which queue to join: Some counterexamples. *Operations Research 34* (1986), 55–62.

Willians, T. M. Nonpreemptive multi-server priority queues. *The Journal of Operational Research Society 31* (1980), 1105–1107.

Winston, W. *Optimal Operation of Congestion Systems with Heterogenous Arrivals and Servers*. PhD thesis, Yale University, 1975.

Winston, W. Optimality of the shortest line discipline. *Journal of Applied Probability 14* (1977), 181–189.

Winston, W. L. Assignment of customers to servers in a heterogeneous queueing system with switching. *Operations Research 25* (1977), 469–483.

Wolff, R. W. An upper bound for multi-channel queues. *Journal of Applied Probability 14* (1977), 884–888.

Wolff, R. W. Upper bounds on work in system for multichannel queues. *Journal of Applied Probability 24* (1987), 547–551.

Wu, J.-S., and Chan, W. C. Maximum entropy analysis of multiple-server queueing systems. *The Journal of Operational Research Society 40* (1989), 815–825.

Yum, T.-S. P., and Schwartz, M. The join-biased-queue rule and its application to routing in computer communication networks. *IEEE Transactions on Communications 29* (1981), 505–511.

Zhang, H., Hsu, G., and Wang, R. Strong approximations for multiple channel queues in heavy traffic. *Journal of Applied Probability 27* (1990), 658–670.

Zhang, Z. Analytical results for waiting time and system size distributions in two parallel queueing systems. *SIAM Journal of Applied Mathematics 50* (1990), 1176–1193.

Zhao, Y. *Shortest Queue Models*. PhD thesis, University of Saskatchewan, 1990.

Zhao, Y., and Grassmann, W. K. A numerically stable algorithm for two server queue models. Tech. rep., University of Saskatchewan, 1989.

Zhao, Y., and Grassmann, W. K. The shortest queue model with jockeying. *Naval Research Logistics 37* (1990), 773–787.

Zhao, Y., and Grassmann, W. K. Solution of jockeying problem with general input. Tech. rep., University of Saskatchewan, 1990.

APPENDIX A LITERATURE SURVEY — CONTROL OF MULTIPLE SERVER QUEUES

A.1 Introduction

A.1.1 Queueing systems

Everyone spends/wastes time in queues everyday. Lining up to purchase gasoline for your car, waiting for hamburgers in fast-food restaurants, waiting for photocopying machines, lining up in banks, etc. are common occurrences. Queueing systems show up wherever you are, whatever you do. You are still in a queueing system even though you are not doing anything, because you are waiting to for something to happen.

Thus we have many good reasons to study queueing theory and understand it well.

The first paragraph from Kleinrock's foreword to [Takagi90] is quoted here as a brief history of queueing theory :

Queueing theory was developed by A.K. Erlang in the early years of this century to serve the needs of the telephone engineer. In the mid-1930's, the mathematicians began to dominate the field and advanced it in many directions (not all of which were of practical use). With the advent of World War II, the field of Operations Research absorbed queueing theory along with a host of other systems engineering tools such as game theory, search theory, mathematical programming, optimization theory, decision theory, etc. Unfortunately, operations research failed to fulfill the promise of solving many of industry's systems engineering problems; its models were too simplistic. However, toward the end of the 1950's, we began to see some very successful applications of queueing theory to the emerging computer industry; indeed, the models were capable of predicting performance and of assisting in design procedures. Many of these early models were used by us for multiuser time-shared

comp[illegible]r systems. In the mid-1960's, computer-communication networks yielded to queueing analysis. Since then, data communication and data processing have grown to be among the largest industries on this planet, and the use of queueing methods to carry out performance analysis and design of these complex systems has been enormously successful.

A.1.2 Queueing theory preliminaries

The objective of this subsection is to provide enough queueing theory preliminaries for the rest of this survey. A general queueing system is shown in figure 10.

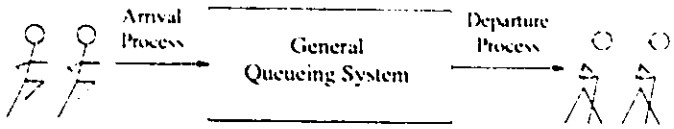

Figure 10 A General Queueing System [Kleinrock75]

The processes of objects entering and leaving the system are called the arrival process and the departure process respectively. The objects go into the system, stay there for some time and then leave that system. The objects could be human beings, job requests, items in supermarkets, etc. The objects are referred to as jobs and customers interchangeably in this survey.

The arrival and departure processes can depend on anything including time and load in the system. They can follow various distributions which can also depend on anything. The number of jobs inside the system is expected to be below a reasonable level, if the arrival process is, on the average, slower than the departure process. The traffic intensity ρ for a single queue is usually defined as λ/μ where λ is a constant average arrival rate and μ is the maximum service rate.

The Kendall-Lee notation (1951) is a standard description of a single waiting line queueing system. The format is

$$\{1\} / \{2\} / \{3\} / \{4\} / \{5\} / \{6\}.$$

All fields are separated by slashes. The first and second fields specify the nature of the arrival process and the service process respectively. Standard abbreviations for these two fields are as follows :

- M = Interarrival / service times are independent, identically distributed (iid) random variables having an exponential distribution.
- D = Interarrival / service times are iid and deterministic.
- E_k = Interarrival or service times are iid Erlang distributed with shape parameter k.
- GI = Interarrival times are iid and governed by some general distribution.
- G = Interarrival / service times follow some general distribution.

The third field specifies the number of servers. The fourth field describes the queue discipline. FCFS and LCFS are the standard abbreviations of "first come first serve" and "last come first serve" disciplines. The fifth field and the sixth field describe the capacity of the queueing system (size of the queue) and the population of the arrival jobs respectively. Whenever the values of the field {4}, {5} and {6} are omitted, they are assumed to be FCFS, infinity, and infinity.

Modification of this notation was suggested for queueing system in which each job chooses its own service team and joins the queue belonging to that team at the time of arrival. The notation is

$$\{1\} / (\{2\} / \{3\} / \{4\} / \{5\})^N / \{6\}$$

which is basically the same as the original Kendall-Lee notation, except that the value of N specifies the number of service teams. Each team forms a queueing subsystem working

independently. The arrivals are divided among to the teams via some algorithms. The service distribution, number of servers, service discipline, and capacity of each queueing subsystem is described by the fields {2}, {3}, {4}, and {5} respectively.

A.1.3 The multiple server queueing system

A small subset of queueing systems, multiple server queueing systems, is the subject of this survey. An abstract diagram of such a queueing system is shown in figure 11. This system is an essential element of more complex queueing systems. This general setting for considering queues is introduced in this survey for the first time by the author. It allows for a methodical way of classifying work done by many researchers.

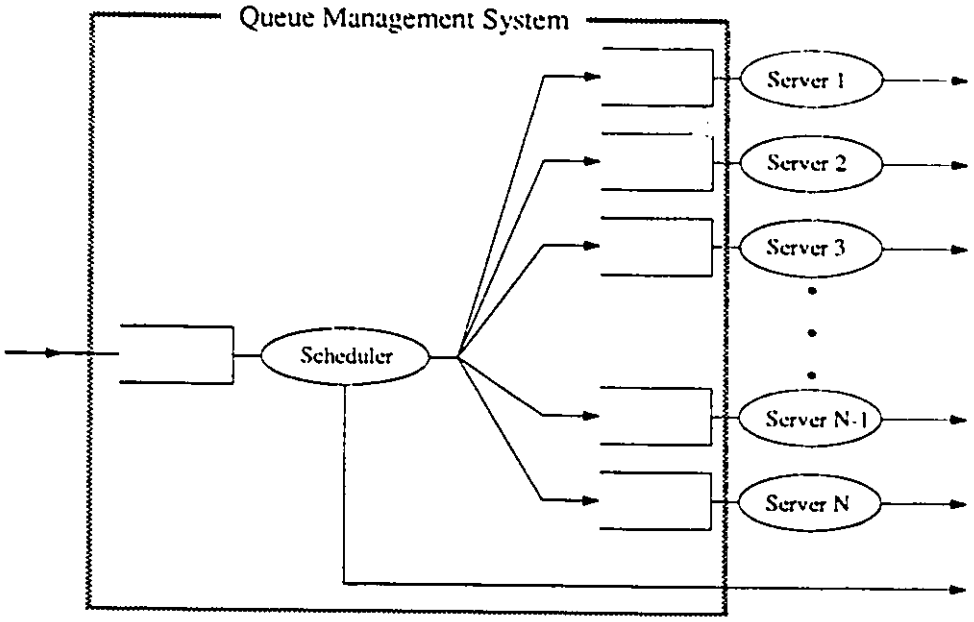

Figure 11 A General Multiple Server Queueing System

Each server in a general multiple server queueing system can represent a team of individuals or another general multiple server queueing system, so that the general system can be defined recursively. However, for simplicity, we assume each server represents

only one individual in this survey unless specified. All servers operate independently in parallel and can do the same job[17]. Their service time distributions may be different.

Let queue 0 be the queue of the scheduler, queue 1 be the queue of server 1, . . . , queue N be the queue of server N.

Jobs of different priority classes are coming from the left hand side. Jobs may have to wait at queue 0 for some reason. For example, the scheduler may be slow or all other queues may be full.

Job populations and the sizes of all queues may be finite or infinite.

The scheduler can choose any job from queue 0 and assign the job to queue i, i = 1, . . . , N according to some rule. The system can refuse to provide service to any job and the rejected jobs bypass all servers in this case.

The interarrival time distributions (of different priority classes) and service time distributions (of different servers and scheduler) may be load dependent or time dependent.

The queue management system can control anything inside the block. It can switch any job from any queue to any other queue, preempt any job in service, and control the capacity of all queues etc. Everything else, such as jockeying rates and arrival rates of different classes of jobs and the scheduler's service time distribution, are arbitrary.

Jobs / customers inside the system look passive when the management system makes all the decisions. Jobs / customers are not passive when they make their own decisions. Both situations can look identical to an external observer.

A.1.4 Objective of control — social versus individual

The queue management system controls everything inside the system for some

[17] Note that even though different servers are doing different kinds of jobs, the shared buffer (queue 0) may still be used. Some systems have limited and expensive capacity, so different kinds of jobs must be queued together for better management of the system [Chang87].

objectives. Objectives can be roughly classified into social objectives and individual objectives. A comparison of these objectives was studied in [Knudsen72, Bell83]. Much research has been done on the control of queues but very little [Hlynka91] focused on the individual objective control.

A social objective is an objective that will optimize a performance index of the system in an overall sense. It may be considered as the objective of a system manager or employer. The manager does not care who gets the better service. The only thing the manager cares about is the service quality averaged over all customers and the overall profit. Here are some examples of social objectives :

1. Maximize servers' utilization — The employers want to optimize their gain from their employees.
2. Minimize servers' wage — The employers want to minimize the amount of time for which they must pay the servers.
3. Minimize average delay / holding cost — If the delay is too long, some customers may go away to be served somewhere else. That costs the company. The delay cost can be measured as response time, queue lengths, dollars per waiting minute, etc. Variances of delay or holding cost should also be minimized to keep a fair system.
4. Maximize reliability — The manager wants to minimize the number of complaints from the customers. If the system is not reliable, the customers might not come back next time.
5. Maximize throughput — The manager wants as many customers as possible to pass through the system.
6. Maximize availability — If the system is always full, the customers may go somewhere else.
7. Fairness of resources allocation — This is sometimes referred to local balancing.

8. Q-factor — This performance index was introduced in [Wang85] to compare various control strategies.

An individual objective is an objective that will optimize a performance index of an individual job. Here are some examples :

1. Minimize or maximize sojourn time / delay cost — A particular customer wants to maximize or minimize its time in the system, depending on whether it is desirable or undesirable to stay in the system.
2. Getting the "best" service — A particular customer may find that one of the servers works faster, is more friendly, is more reliable, or provides better service. The customer may prefer to be served by that server and wait longer.

Example : Consider a queueing system with two servers where customers line up in a FCFS combined line[18]. One of the servers is very slow because it is a new server. The other server is quite fast. Normally, customers will prefer the faster server. So the customer at the head of the line will wait until the fast server is available, even if the slower server is idle. Because the line discipline is FCFS, the line is blocked even though the second customer wants to join the slow server. The behavior of the first customer satisfies the individual objective, because its individual sojourn time is minimized. But it is not socially optimal, because the customers do not fully utilize the slower server [Daduna87]. Thus individual and social objectives may conflict with each other. Additionally, it is also possible for one social objective to conflict with another social objective. A well known example is the conflict between servers' utilization and average waiting time [Lazar84].

---

[18] This is a special case of the general multiple server queueing system described earlier when the capacity of each of queue 1 to queue N is only one and N = 2.

A.1.5 Different assumptions

The resulting optimal policy would be completely different with different assumptions. Here are some assumptions that are sometimes used :

1. Servers' statuses — 'whether the servers are idle or busy' are known.
2. Scheduler's assignment sequence — 'the last server that was scheduled' is known.
3. Homogeneous servers — the service distributions of all servers are assumed to be equal.
4. Homogeneous customers — only one class of customers.
5. Queue lengths are always available.
6. Service disciplines — first come first serve (FCFS), last come first serve with preemptive resume (LCFS / PR), last come first serve with non-preemptive resume (LCFS / NPER), round robin (RR), etc.
7. Service and interarrival time distributions — these distributions are sometimes available through some statistical technique [Dave85]. In addition, they are assumed to be time homogeneous.

Different assumptions under different policies will be discussed later in this survey.

A.1.6 Layout

The structure of the remainder of this survey is as follows. In section A.2, we will discuss the general properties of multiple server queues. In section A.3, techniques for the analysis of multiple server queueing systems under different control policies will be introduced. Control policies for multiple server queueing systems are discussed in section A.4. In section A.5, some actual systems that can be modelled by the general multiple server queueing system are given.

A.2 General Properties of Multiple Server Queues

Special cases of the general multiple server queueing systems are mainly classified into parallel queueing systems and combined queueing systems. Few studies have involved general multiple server queueing systems.

A.2.1 Definition of Parallel Queues

A parallel queueing system, as shown in figure 12, is a special case of the general multiple server queueing system described in the introduction. Its special characteristics are

1. the capacity of queue 0 is one and
2. the scheduler is infinitely fast.

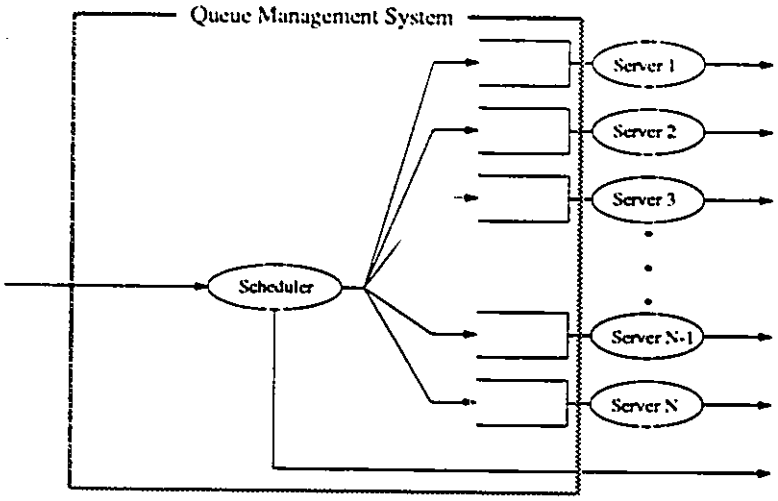

Figure 12 A General Parallel Queueing System

A.2.2 Definition of Combined Queues

A combined queueing system, as shown in figure 13, is a special case with no waiting room in front of server i, for all i=1,2, . . . ,N. The only difference between parallel queueing and combined queueing systems is the order of scheduling and waiting.

Jobs in parallel queueing systems are assigned to servers *before* waiting in a queue, but jobs in combined queue systems are assigned to servers *after* waiting in a queue.

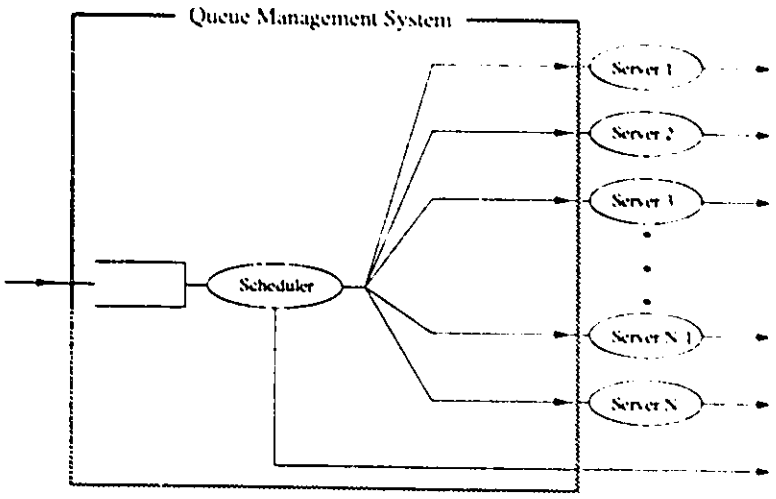

Figure 13 A General Combined Queueing System

Most research has examined job assignment policies of the scheduler to minimize the delay cost, and to maximize the server utilization and throughput.

The queueing discipline of queue 0 is assumed to be FCFS and non-preemptive unless otherwise specified.

A.2.3 Parallel Queues versus Combined Queues

Combined queues are traditionally preferable to parallel queues. Combined queue designs are generally regarded as being more efficient. The following are some advantages of combined queue designs over parallel queue designs :

1. all servers in the systems are fully utilized.
2. load balance is ensured.
3. a limited total system capacity can be effectively managed.

However, the environment sometimes does not allow us any choice of system design. Parallel queues may form naturally. Moreover, the shared buffer (queue 0) is not allowed in some systems. Thus the jobs must join queues in parallel. Examples are :

1. Customers in a supermarket form parallel queues in front of each cashier and no scheduler forces the customers to do so.
2. Different classes of jobs are served by different servers. This kind of design is simple and provides good performance for higher class jobs. A practical example of this is the CMS system in the computer center of the University of Windsor. There are two central processing units (CPU) in the computer system. One is for academic purposes, another is for university administration.
3. Two different barber shops located next to each other have their own queues. No shared buffer exists in this case.

In addition, parallel queue design is sometimes preferable to non-parallel queue design, even if the system designers have choices. The following are some advantages of such parallel queueing systems [Rothkopf87], especially for queues of people :

1. Servers' responsibility— the employer will notice which server is the slowest, because a longer line can be found, on the average, in front of the slowest server. The slow server should work faster. Otherwise he/she may be fired. Each server is responsible for its own queue. There is no such responsibility in combined queueing systems.
2. Shorter appearance — Parallel queues look shorter to ordinary customers. That is why people form parallel queues rather than a combined queue in the supermarket example.
3. Freedom — customers are allowed to choose their own servers.

A.3 Analysis Techniques for Multiple Server Queues

Analysis of the general multiple server queueing system is not easy. Advanced techniques, which are required to analyze, and therefore understand, the performances of various control strategies, are reviewed in this section.

A.3.1 Traditional Techniques

A.3.1.1 System Equilibrium Problems in queueing theory are traditionally solved by modelling the queueing systems with continuous-time discrete-state Markov processes and by assuming that the systems are in equilibrium. The following is a typical solution procedure[19] :

1. Define the system states.
2. Draw the transition rate diagram for the respective queueing system.
3. Write down the global balance equations for the transition rate diagram by assuming system equilibrium.
4. Solve the set of equations for the steady state probability.
5. Predict the behavior of the system by using the steady state probability distribution.

The usual assumptions are

1. exponential interarrival time,
2. exponential service time, and
3. system equilibrium.

But these assumptions are not necessarily appropriate. The interarrival and service time distributions may be deterministic, Erlangian, Coxian, or any other probability

[19] A standard introduction of this technique is presented in [Kleinrock75].

distribution. Additionally, the system may not be in equilibrium, because there may be 'rush hours' in reality. (i.e. the arrival rate may sometimes be higher than service rate.)

Furthermore, difficulty in solving the set of global balance equations, either analytically or numerically, is an obstacle. Techniques such as the use of generating functions [Kleinrock71], recursive solutions [Herzog75, Brandwajn79, Chow79] and matrix-geometric solutions [Neuts81, Gertsbakh84, Elsayed85, Ramaswami86] were developed to solve the set of equations. Specialized techniques for two-dimensional Markov models have also been used [Cohen83, Fayolle82, Blanc84].

A.3.1.2 Discrete Approximations For general interarrival and service times, the solution steps described in section 2.1.1 cannot be applied. One way to overcome this is to use discrete approximation [Kleinrock76]. The queueing systems are modelled by discrete-time discrete-state Markov processes. The system time is divided into a number of small quanta. System state transitions are only allowed between consecutive quanta. However, there is difficulty in defining the length of each quantum. The quanta could be small equal length quanta. They may also be time intervals between system epochs such as arrivals and departures. For example, the interarrival time was chosen as the quantum for modelling of M/G/1 queueing systems [Kleinrock75]. A device [Lippman75] was introduced in attempting to specify the form of an optimal control policy, which describes the system by a series of transitions to enlarge the standard set of decision epochs.

A.3.1.3 Simulations Simulation is usually considered to be the last resort in studying system behavior, when no analytical tool is available. This is because simulation programs are not guaranteed to be "correct." (i.e. a simulation program may not actually simulate the system that the programmer wants to simulate.) Moreover, simulation of random events requires a random number generator, but the "randomness" of such

generators is also not guaranteed. Also simulation handles only special cases. It may not give insight available from mathematical analysis. These are some of the disadvantages of simulation. A survey of steady-state simulation has been carried out recently [Pawlikowski90].

A.3.2 Information Theoretic Approximations

Entropy is a measure of uncertainty, randomness, or fuzziness. This measure H was defined [Shannon48] as

$$H(f) = -E[\ln(f(X))]$$

where X is a random variable in space A with a probability density function f. The principle of maximum entropy was established [Jayne57] and was applied to approximate the behaviors of queueing systems. These approximations are called information theoretic approximations. Some characteristics of these approximations such as correctness [Shore80], existence [Csiszàr75] and error estimation [Shore82] were discussed.

The concept of entropy was first introduced by Clausius (1850) in the area of thermodynamics. Subsequent developments were mainly in that area. There was not much work before Shannon's breakthrough on information theory in 1948. Shannon applied the concept of entropy to study typical sequences and solved a number of problems in coding theory and data transmission. In 1957, Jaynes applied the idea again by introducing the principle of maximum entropy to solve problems in statistical mechanics. One of the earliest applications of entropy in queueing theory was carried out in 1970 [Ferdinand70]. Recent publications in this area include [Papoulis90, Kouvatsos89_1, Kouvatsos89_2, Wu89].

A.3.3 Diffusion Approximations

A.3.3.1 Heavy Traffic Approximations The idea of heavy traffic approximations is to simplify the system by considering an extreme case where $\rho \simeq 1$ (but remains strictly less than one to preserve stability, where ρ is the traffic intensity as defined in the introduction.) The method usually provides bounds for some performance measures such as the average wait.

One of the earliest studies of such approximations appears in 1961 [Kingman61_2] and recent studies include [Courcoubetis88, Ramachandran89].

A.3.3.2 Fluid Approximations Fluid approximation, which is also called "first-order" approximation, approximates performance measures by their averages (first moments). Higher-order moments of those measures are not considered. Thus, the flow of jobs through a system is approximated by the flow of an incompressible fluid through a pipe or a funnel where the flow is continuous and deterministic.

Let μ(t) be a time-dependent decreasing rate of the number of jobs when there is no new arrival and λ(t) be a time-dependent arrival rate. Consider an arrival process with arrival rate λ(t) and a service process with service rate μ(t) to a queueing system. μ(t) depends on λ(t), since the servers of the system can be idle. In order to make λ(t) and μ(t) independent, the traffic intensity ρ is assumed to be very high. Then the number of jobs, N(t), staying in the system can be obtained by solving the following differential equation :

$$\frac{dN(t)}{dt} = \lambda(t) - \mu(t), \quad where \quad N(0) = N_0.$$

This allows us to approximate system behavior before and after a rush hour. (i.e. $\rho - 1 > \epsilon$, for some $\epsilon > 0$.) It should be noted that the systems are originally discrete state systems and they are approximated by continuous state systems.

Fluid approximation predicts that the number of jobs N(t) equals zero in the long run, if $\lambda(t) = \mu(t)$. However, traditional queueing theory, supported by observing systems, does not agree with this "first-order" approximation for stochastic arrival and service processes.

Diffusion approximation ("second-order" approximation) is an extension of fluid approximation, which approximates the arrival and service distribution by normal (Gaussian) distributions with their means and variances. Diffusion approximation overcomes the difficulty of approximating system behavior when $\lambda(t)$ is close to $\mu(t)$.

Early studies of diffusion in the analysis of queueing systems were undertaken in the 60's [Iglehart65, Gaver68]. Diffusion was subsequently applied successfully to analyze various queueing systems [Foschini78, Foschini82, Giorno86, Giorno87, Kella90]. Details of heavy traffic, fluid approximations, and diffusion approximations were discussed in [Kleinrock76].

A.3.4 Light Traffic Theory

As discussed in the last section, the analysis of system behavior may be easier under the heavy traffic assumption. One may wonder if the analysis is also easier under light traffic conditions. The earliest study [Benes65] of queueing systems in light traffic is concerned with the modeling of telephone switching systems. The subsequent developments are concerned with estimation of queue length distributions and sojourn time [Reiman89_3]. A formal theory — light traffic theory [Reiman89_3] has recently been developed for estimation of general performance indices of queueing systems under light traffic conditions.

Let $f(\lambda)$ be some quantity of interest where λ is the arrival rate. The theory can be applied to investigate the behavior of $f(\lambda)$ when λ is near zero. Values of $f^{(n)}(0)$ can be

obtained for some positive n and hence the value of $f(\lambda)$ can also be obtained for small λ by using Taylor's formula [Reiman89_3]. When λ is not small, values of $f(\lambda)$ can also be estimated by interpolation for λ between 0 and λ^*, if some information about $f(\lambda)$ at some value $\lambda^* > 0$ is known.

A.3.5 Other Analysis Techniques for Multiple Server Queues

Other powerful techniques follow:

1. Dynamic Programming — This technique was presented in [Howard60] in detail and was applied to define an optimal control strategy in [Krishnan90].
2. Combinatorics — this is a very broad technique. No particular method in combinatorics can be applied to solve general queueing problems. However, results from combinatorics have been used to solve queueing problems since 1967 [Takács67]. One of these applications was reviewed in [Kleinrock75]. Recent applications of this technique include [Sato83, Ohta85]
3. Approximation by Another System — The idea of this technique is simple. In some cases, the behavior of one system is close to the behavior of another system, but the performance analysis of one system is easier than the other. Thus, the performance of the second system can be approximated by the performance of the first system. This technique was applied to approximate the behavior of the JSQ model by the behavior of the M/M/2 queueing model [Nelson89]. Another example is [Wolff87].

A.4 Control of Multiple Server Queues

A.4.1 The Control Model

A formal definition of a control policy has been given for multi-scheduler multiple server systems [Boel89]. The definition is rewritten here for single scheduler multiple server systems with the single job priority assumption.

Let N be the number of servers in the system.

Let t be the system epoch number where $t \geq 0$. The value of t is increased by one at each system epoch such as a job arrival or a service completion.

Let

$$X_t = \begin{bmatrix} x_{0,0} & x_{1,0} & \cdots & x_{N,0} \\ x_{0,1} & x_{1,1} & \cdots & x_{N,1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{0,t} & x_{1,t} & \cdots & x_{N,t} \end{bmatrix}$$

be the state matrix of the system where $x_{i,\tau}$ is the length of queue i including the customer in service at time τ for i = 0,1, . . . ,N and τ = 0,1, . . . ,t.

Let e_t be the event matrix where row τ represents what happened at epoch τ.

Then the control function G can be defined as follows :

$$D_{t+1} = G(e_t, X_t, D_t)$$

where D_t is the decision matrix with row τ representing the decision made at epoch τ. If the function G is independent of X_t, then the strategy corresponding to G is called a static strategy, otherwise it is called a dynamic strategy. The objective of most research in this area is to find the optimal function G.

For example,

1. row τ of e_t = [5,0,0, . . . ,0] means there is an arrival of 5 jobs at epoch τ.
2. row τ of e_t = [0,−1,0, . . . ,0] means there is a service completion by server 1 at epoch τ.
3. row τ of D_t = [0,1,−1, . . . ,0] means a job is jockeying from queue 2 to queue 1 at epoch τ.

Control strategies of parallel queues have more variations than control strategies of non-parallel queues and are consequently more complex. Therefore, control of parallel queues will be reviewed after the review of control of non-parallel queues.

A.4.2 Control of Non-Parallel Queues

The control of combined queueing systems is reviewed first, the control of the less common general multiple server queueing system is reviewed at the end of this section.

A.4.2.1 Classical Strategies If an arriving job finds all servers busy, it joins the combined queue and waits until any server is available. If all N servers are free, the arrival job is assigned randomly to one of the N servers according to some probability distribution $[p_1, p_2, \ldots, p_N]$. Suppose more than one and less than N servers are free. Say k of them are free. Labelled them as $a_1, a_2, \ldots, a_k$. The arrival job is assigned to a_i with probability

$$P\left(A_i \mid \bigcup_{j=1}^{k} A_j\right) = \frac{p_{a_i}}{\sum_{j=1}^{k} p_{a_j}}$$

where A_i is the event that the job is assigned to server a_i, i = 1,2, . . . ,k. The probability distribution was taken to be a uniform distribution in the earliest design [Gumbel60]. If the servers are homogeneous, the queueing system degenerates to a G/G/N system[20]. Non-uniform distributions were suggested [Krishnamoorthi63] for heterogenous server systems. The classical strategy is also referred to as the stochastic assignment rule.

The earliest design was modified by assigning jobs to servers in a cyclic manner and this model was analyzed [Wolff87].

The classical strategy is a static policy. No up-to-date information about the system is required to make the job assignment decision. This is not an optimal policy [Lin82, Lin84], if the current status of the systems is available. Nevertheless, this policy is good

---

[20] There is a great deal of research on G/G/N systems such as [Brumelle71, Wolff77] and their variations (including priority systems [Gail88] and systems where customers can refuse to join the queue [Knudsen72].) Research for these systems is still active, see [Parthasarathy89, Knessl90].

for systems where collecting information is costly. The classical strategy is also simple for performance analysis.

A suggestion was made [Daduna87] for a heterogenous server system such that if all servers are available, jobs in queue 0 are scheduled to the fastest server. This idea led to the threshold strategy, described next, which takes further advantage of the fast server.

A.4.2.2 Threshold Strategy The roots of this strategy can be found in [Krishnamoorthi63, Rubinovitch85], but the idea was formally analyzed and proved to be the optimal strategy [Larsen83, Lin84, Reiman89_2] later. The strategy assumes that information on the servers' statuses and number of jobs in the systems is available.

Assume the service rates of the servers are ordered such that server 1 is the fastest, server 2 is the second fastest, . . . , and server N is the slowest.

Let L be the length of queue 0.

Let the thresholds be $L_0^*, L_1^*, \ldots, L_N^*$ such that $0 = L_0^* < L_1^* < \ldots < L_N^*$.

The "threshold strategy" was defined for scheduling jobs to N servers by the following algorithm :

> Find i such that $L_{i-1}^* \leq L < L_i^*$.
>
> Find k such that servers 1,2, . . . ,k-1 are all busy while server k is idle.
>
> If k ≤ i, assign a job in queue 0 to server k.
>
> else hold on until any of the servers 1,2, . . . , i is available.

Different objectives will lead to different thresholds (policies.) Computation of thresholds for minimizing the mean number of jobs in the system and total flow time can be found in [Larsen83, Agrawala84] respectively.

Thresholds for priority systems have been studied [Reiman89_1] where the queueing discipline of queue 0 is no longer FCFS.

This strategy does not refuse any job request because the size of buffer (queue 0) is infinite. If the buffer size is finite, the assignment of jobs will depend on the reward of different jobs [Courcoubetis88].

A.4.2.3 Adding / Removing a Group of Servers The idea of this policy [Bell75, Huang77, Bell80, Rhee90], and the assumptions of available knowledge made in it, are similar to the threshold policy, except that the turn on / off cost of this policy is no longer zero.

This type of policy reduces the total wages paid to servers. However, it involves some set up costs and shut down costs, because

1. for human systems, the servers may not be able to keep up with the usual load after a long layoff period. In that case, the employer may have to retrain the employee.
2. for computer systems, overhead is required to communicate with the processes or peripheral devices.
3. for production lines, starting up a machine involves set up costs and may affect quality.

As a final note to the review of combined queueing systems, the threshold strategy has been proven to be optimal when the cost of turning a server on / off is zero [Larsen83, Lin84, Reiman89_2]. However, no conclusions have been reached for systems where turning a server on / off is costly. Thus, some of the fundamental properties [Rosberg90] of optimal strategies are still being studied.

A.4.2.4 The General Multiple Server Queueing System — "Observing Before Joining" Only one general multiple server system related study has been carried out

[Hlynka91]. The strategy used in the study is called "observing before joining" which minimizes the incurred cost to an individual customer. The information available to a new customer entering the system is limited to the length of each queue. Any customer can stay in a shared buffer (queue 0) to observe the queueing system and infer the respective cost, if that customer joins a particular queue. The observing customers can then join the queue that costs the least. Nevertheless, observing the system will have a cost to the observing customers. Thus, customers should make their decisions and join queues as soon as possible. There is a general rule, "the longer the observation period, the better the cost estimate is." Consequently, a good method is required to estimate the cost in a minimum observing time. The study showed that some strategies, on average, do give a shorter sojourn time provided there are very few customers observing before joining, and there are significant differences in the service rates.

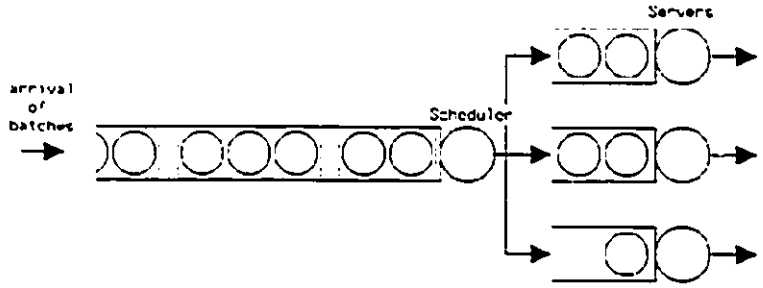

Figure 14 An Example of a Fork/Join System with 3 Servers.

A fork/join system [Flatto84, Baccelli85, Flatto85, Chang87, Courcoubetis87, Duda87, Nelson88_1, Nelson88_2, Towsley90, Zhang90] is close to but not a special case of the general multiple server queueing system, because it violates the assumption that all jobs can be served by any server independently. The arrival of jobs in the fork/join system are in bulk. Batches of jobs are split to different servers by the scheduler and the jobs are served in parallel. The service processes of the jobs are dependent. Consequently, the issues in fork/join systems are not discussed further here.

A.4.3 Control of Parallel Queues

Some control strategies can be "moved" from the previous section to this section. As pointed out before, the only difference between combined queue control and parallel queue control is the order of scheduling and waiting. As a result, the control strategies of parallel queues can be considered as the dual of that of combined queues.

A.4.3.1 Classical Strategies These strategies are similar to those described in last section except that the jobs will be immediately assigned to different servers when they arrive. Jobs are assigned to servers stochastically [Chow79]. Conditional probabilities are not required, provided that the sizes of all waiting rooms are infinite, which is a common assumption. If the sizes of some waiting rooms are finite and some of the queues are full, then conditional probability is applied. This is equivalent to the busy server situation.

This stochastic assignment is a static control strategy. It requires the least effort to collect up-to-date information.

The following are some minor modifications of this simple stochastic policy.

1. Jobs can be assigned to different servers in a cyclic manner as described in previous section. This is referred to as cyclic splitting [Wang85] and round robin [Ephremides80] in the literature. This is also a static control policy.
2. Customers can be assigned to different queues with autonomous traffic control [Lemoine75]. This is similar to traffic light controls for vehicles on the roads.
3. The assignment probabilities can be updated periodically. In this case, the policy becomes a dynamic policy [Bonomi90].

A.4.3.2 Jockeying Transferring jobs from some queues to others is called jockeying. Jobs are usually transferred from a long queue to shorter queues to balance the load of

all servers. In an extreme version of jockeying, jobs in service can also switch servers. This strategy can also be applied to combined queueing systems.

In order to make a meaningful jockeying action, up-to-date information about the systems, such as queue lengths of all queues, is required. Jockeying is thus a dynamic strategy.

Jockeying is considered as an action that minimizes individual sojourn time. Jockeying is also ideal to allocate the jobs uniformly to different servers. Therefore, the objectives of this strategy can be considered as both social (local balance) and individual. Because jockeying and JSQ have the property that they distribute the jobs uniformly to all servers, they are often used together.

Early research was done for two server systems [Haight58, Glazer58, Koenigsberg66, Davis77]. Later research was done for k-server systems with finite buffers [Disney71, Elsayed85] where $k \geq 2$. Matrix-geometric type solutions [Kao90] and non-recursive solutions [Zhao90_1] were obtained. However, all research mentioned above assumes a Poisson arrival process. The jockeying problem with a general arrival process was recently studied [Zhao90_2]. Some modification of jockeying was made to form partial jockeying which was also recently studied [Wang90].

A.4.3.3 "Join the Shortest Queue" (JSQ) Strategy The strategy of scheduling jobs to the server with the shortest queue, is called "join the shortest queue" strategy. This queueing model was originally created to model customers' behavior in systems made up of humans [Koenigsberg66]. The objective of this strategy was traditionally considered as minimizing individual sojourn time. However, this strategy was used later for load balancing [Wang85, Boel89] which is a social objective. The information required to

make the decision is knowledge of the shortest queue. This information should be up-to-date. Therefore, JSQ is a dynamic strategy.

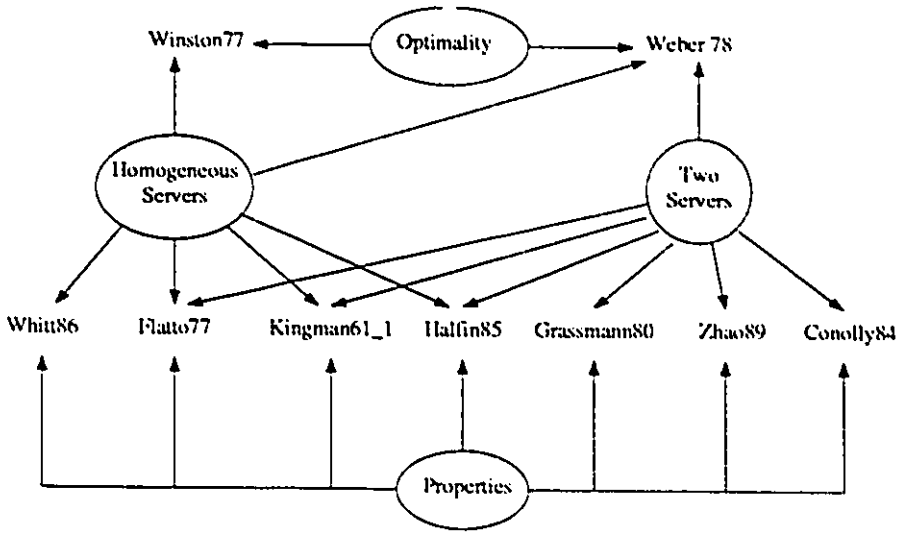

Figure 15 JSQ Research Relations

The JSQ strategy has been studied for more than thirty years since 1958 [Haight58]. It was proved to be optimal for homogenous server systems [Winston77_1, Weber78]. Its properties were studied in many research papers [Kingman61_1, Flatto77_2, Grassmann80, Conolly84, Halfin85, Whitt86, Blanc87, Rao87, Gubner89, Nelson89, Zhao89, Adan91], but no analytic queue length distributions for the queueing models has been found in the heterogeneous server case. A network relation of some research is shown in figure 15.

JSQ has been modified. Here are some of the attempts.

Generalized JSQ [Foschini77] — In this strategy, an arriving job is assigned to the server with the least expected sojourn time. It was compared with a socially optimal static rule described in section 3.3.1. It was identified as an individually optimal dynamic rule [Stidham85, Krishnan90].

Join the expected shortest queue [Ephremides80] — The first job is assigned to the first queue and other jobs are assigned to the queue with the shortest expected length. This rule was applied to a parallel two server queueing system. However, this strategy was inferior to JSQ and this result was mentioned in the research paper [Ephremides80] where "join the expected shortest queue" strategy was defined.

Join-Biased-Queue Rule [Yum81] — This is a hybridized design of JSQ and stochastic routing. The root of this strategy is delta-routing [Rudin76]. Let γ_1, γ_2 be the basic (background) Poisson arrival rates of jobs to queue 1 and 2 and γ be the biased arrival rate. Let q_1, q_2 be the lengths of queue 1 and 2; and let λ_1, λ_2 be the resultant arrival rate. The join-biased-queue rule for two-server queueing system was defined as

$$(\lambda_1, \lambda_2) = \begin{cases} (\gamma_1 + \gamma, \gamma_2) & \text{when } q_1 < q_2 + \delta \\ (\gamma_1, \gamma_2 + \gamma) & \text{when } q_1 > q_2 + \delta \\ (\gamma_1 + \beta\gamma, \gamma_2 + (1-\beta)\gamma) & \text{when } q_1 = q_2 + \delta. \end{cases}$$

here δ is an integer, which represents the bias level. $0 \leq \beta < 1$ is the priori probability for routing the biased arrival stream to queue 1 when $q_1 = q_2 + \delta$. The JSQ strategy is a special case of this strategy, when γ_1, γ_2 are set to zero and β is set to 1/2. The classical stochastic strategy is also a special case, when γ is set to zero. Another strategy called JSQ-BS routing rule is a further modification of this strategy and applied in network design [Yum81].

Separable Rule [Krishnan90] — This is the most recent strategy. Jobs are assigned to the queue where the minimum relative cost of transition is incurred. This strategy was compared with generalized JSQ and reported to be "practically always better than generalized JSQ."

A.4.3.4 Overflow Routing Let $L = (L_1, L_2, \ldots, L_N)$ and $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_N)$ respectively be a vector of thresholds and a vector of randomization probabilities. Let

q_i be the length of queue i. Then an overflow routing policy was defined [Rosberg89, Boel89] as follows :

i. If $q_1 < L_1 - 1$, then the job is routed to server 1 with probability 1. If $q_1 = L_1 - 1$, the job is routed to server 1 with probability α_1.

ii. For i, $2 \le i \le N-1$, only the overflow stream from servers 1,2, . . . , i-1, may be routed to server i. If $q_i < L_i - 1$, then the overflow job is routed to server i with probability 1. If $q_i = L_i - 1$, the job is routed to server i with probability α_i.

iii. The overflow stream from servers 1,2, . . . , N-1, is routed to server N.

This is called overflow routing, because the sizes of waiting rooms of queue 1 to queue N-1 can be considered as $L_1, L_2, \ldots, L_{N-1}$.

To make the scheduling decision, up-to-date queue lengths of the system must be known. So this is a dynamic strategy.

The values of L and α are to be determined by setting different objectives. Three implementations of this rule were suggested and analyzed [Rosberg89]. They were overflow with optimal probabilities, overflow routing with proportional thresholds, and overflow routing with shortest wait and proportional thresholds.

Many dynamic strategies such as stochastic routing with periodic review, jockeying, JSQ, and overflow routing have been proposed. However, few studies have been done on the general properties of dynamic strategies [Hall71]. These general properties can help to understand the relationships among all these dynamic strategies. A study of strategy quality was undertaken [Rudin76] for communication networks.

A.5 Applications

A.5.1 Queueing Model of Pseudo Multiprocessing

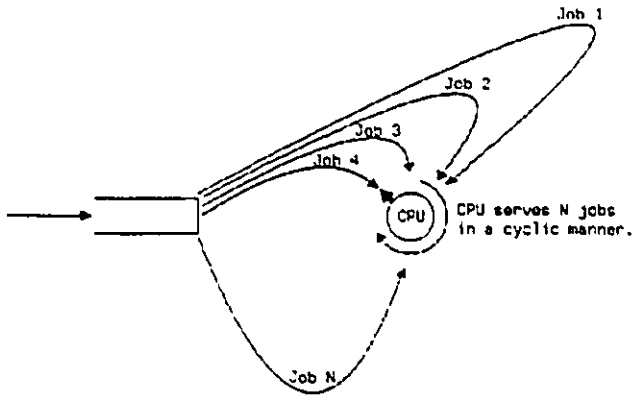

Figure 16 Pseudo Multiprocessing System

Operating system designers find that an ordinary process takes most of its time doing input and output. As a result, the CPU will spent most of the time idle waiting for the I/O completion. In order to fully utilize the CPU, a pseudo multiprocessing design was proposed where a number (N) of jobs are virtually processed by their own CPUs. However, in fact, a CPU can only execute one instruction at a time. One way to solve this problem is to divide the CPU time into a number of small quanta. Jobs are allocated a set of quanta so that they are served by the CPU in a cyclic manner (see figure 16). But the number N cannot be arbitrarily large, because a very large N will sharply decrease the system throughput. Thus N is usually fixed.

In order to analyze this system, the job switching time is assumed to be zero. As the length of each quantum approaches zero, the system will become an ideal process sharing system which can be modelled by a G/G/N queueing system.

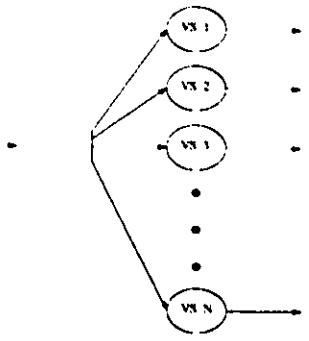

Figure 17 G/G/N Queueing System

Figure 17 shows a stream of jobs coming from the left. In order to maintain an acceptable throughput, the degree of multiprogramming is kept as N. Thus, N jobs are processed by N virtual servers. Any additional arriving jobs must join the combined queue, if the N virtual servers are all occupied. This G/G/N is classified as a combined queueing system with the classical stochastic assignment strategy in this survey.

A.5.2 Repairman Model

A maintenance department of a company hires a manager and two repairmen working independently. The manager (job dispatcher) of this department assigns the orders from other departments to either of the repairmen, whenever there is any machine failure reported. After the machines have been fixed, the machines are sent back to work in the machine pool. The machines work in their departments (machine pool) until they break down again. The machines move around this maintenance cycle as shown in figure 18.

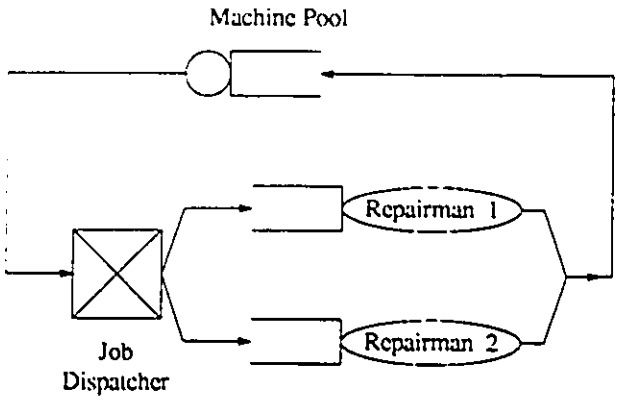

Figure 18 Repairman Model

This system can be modelled by a parallel queueing system with a finite job population. A usual responsibility of the manager (job dispatcher) is to assign the orders to the two repairmen in such a way that the average turn around time is minimized.

A.5.3 Applications in Distributed Operating Systems

In a distributed operating system, different computing units can have different loads at different times. However, leaving a group of computing units idle while keeping another group busy all of the time is not a good way of managing resources. The designer might want to balance the load among all computing units.

Load balancing in order to share the computing power has already been a major design issue. Before going into further detail, a brief introduction to one typical hierarchical design [Tanenbaum85] will be given below.

In this design, a group of computing units is managed by a particular computing unit. Their relationship is similar to a group of workers and their boss. In addition, a group of 'bosses' is managed by a 'dean' computing unit and so forth.

Considering the figure 19 below. There is a queue lining up outside each computing unit. The maximum degree of multiprogramming for each computing unit is assumed to be finite. The queueing discipline inside the system and the average service rate of

each computing unit is not important here. Once the job enters the computing unit, it is considered as having finished its service. The 'boss' is just like a dispatcher assigning jobs (processes) to different computing units, including itself.

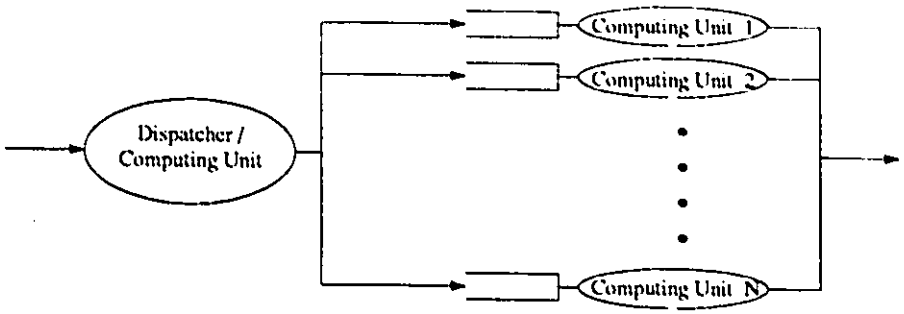

Figure 19 Typical Part of Hierarchical Distributed Operating System

The above system can be modeled by a $M/(G/1)^n$ queueing system.

A.5.4 Applications in Computer Networks

Consider a simple computer network. There are links connecting all nodes except node 2 and node 4. The cost of transmission through any link is assumed to be the same.

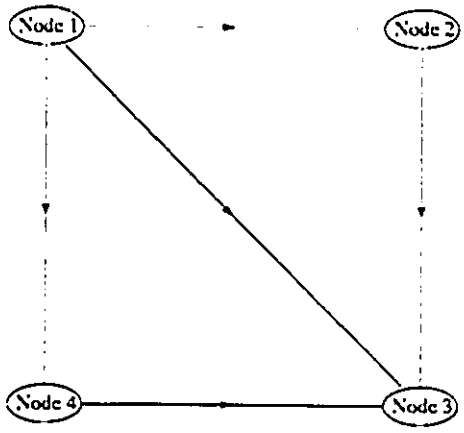

Figure 20 A Simple Network

When node 1 wants to send a big file to node 3, according to the shortest path rule, the file should be transmitted through the link connecting node 1 and node 3. However, it

was shown that the shortest path rule is not the best rule. This rule does not fully utilize other available resources (the route from node 1 to node 2 to node 3 and the route from node 1 to node 4 to node 3). Each route and each packet can be considered as a server and a job respectively. This is a routing and scheduling problem and can be modelled by a three server queueing system [Yum81].

A.6 Concluding Comments

Many control strategies were reviewed for both combined queue and parallel queue designs. There are virtually infinitely many system management strategies. Only major classes have been discussed here. Systems without any waiting room [Winston77_2, Courcoubetis88] (see figure 21) and multiple schedulers [Boel89] are interesting and practical, but they are not our focus. So they have not been discussed here.

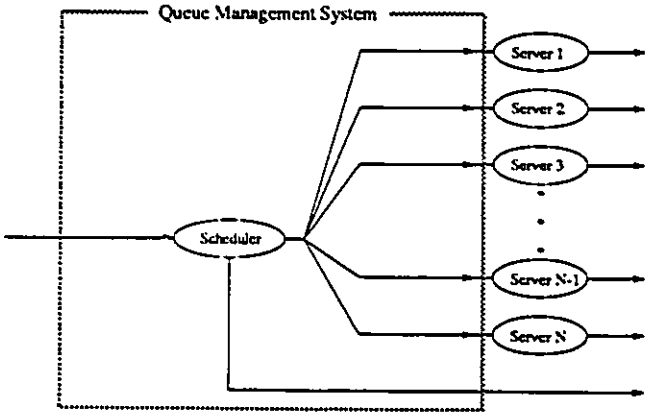

Figure 21 A General Queueing System without Any Waiting Room

As seen in section 4, different optimization criteria will lead to different control policies. Quoting [Boel89], *as control engineering experience suggests, we cannot expect to be able to find an "optimal policy."* What can be achieved is only an acceptable policy. Systems in reality are very complicated, involving many factors. As stated by Rudin [Rudin80], it is dangerous to draw conclusions from the study of different factors

independently and infer that they are also valid when those factors are combined. Thus, even if a strategy is "optimal" in a study, it may not really be a good strategy in general.

From an informal analysis of the strategies discussed in this survey, one can deduce 'a rule of thumb' — the more information available , the better the strategy will be.

As we develop more powerful techniques and tools, future studies will tend to study more general (complicated) systems. More and more restrictive assumptions such as single class arrivals, exponential servers, and FCFS queueing discipline will be removed. In addition, the general multiple server system in figure 11 is just the basic element of some other highly complicated systems. Tandem systems, for example, can be built by using the basic elements. The analysis of such complicated systems is not easy. Researchers in this area still have a long way to go.

APPENDIX B VITA AUCTORIS

Wai-Hang Poon was born in 1966 in **Hong Kong**. He graduated from **Lingnan Dr. Chung Wing Kwong Memorial Middle School** in 1986. From there he went on to the **Hong Kong Baptist College** and studied for a year. Then he transferred to the **University of Windsor** where he obtained a B.Sc. in Mathematics and Computer Science in 1990. This thesis completes the requirements for a Master's degree in Computer Science from the University of Windsor